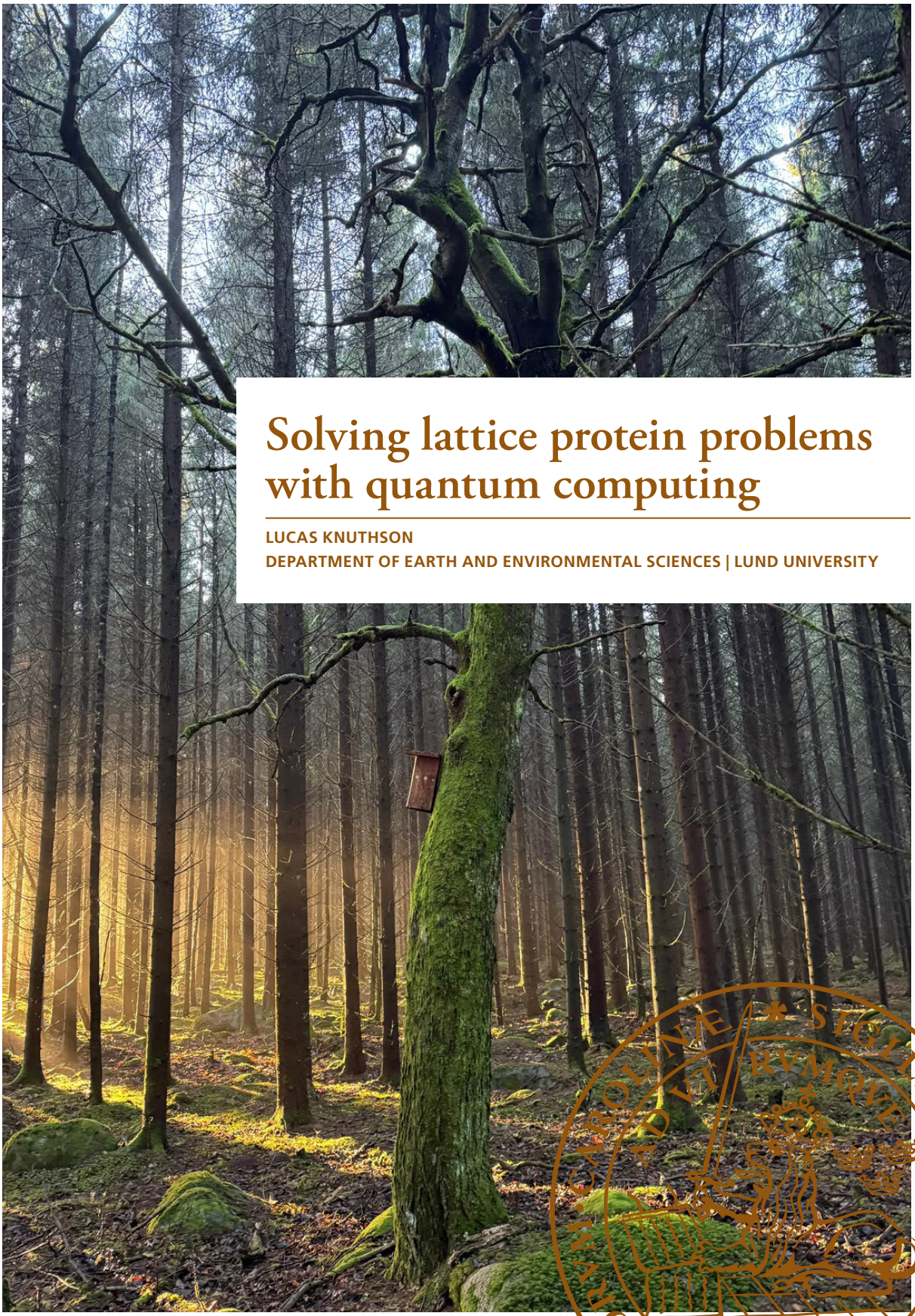
Solving lattice protein problems with quantum computing

Knuthson, Lucas

2026



Solving lattice protein problems with quantum computing

LUCAS KNUTHSON
DEPARTMENT OF EARTH AND ENVIRONMENTAL SCIENCES | LUND UNIVERSITY

Printed by Media-Tryck, Lund 2026 NORDIC SWAN ECOLABEL 3041 0903

Faculty of Science
Computational Science for Health and Environment
Department of Earth and Environmental Sciences

ISBN 978-91-90202-75-3

Solving lattice protein problems with quantum computing

Solving lattice protein problems with quantum computing

by Lucas Knuthson

LUND
UNIVERSITY

Thesis for the degree of Doctor of Philosophy
Thesis advisors: Prof. Anders Irbäck
Faculty opponent: Prof. Cristian Micheletti

To be presented, with the permission of the Faculty of Science of Lund University, for public criticism at Pangea (Sövegatan 12, floor 2) on Thursday, the 17th of September 2026 at 13:15.

Organization
LUND UNIVERSITY

Computational Science for Health and Environment
Department of Earth and Environmental Sciences
Sölvegatan 12, 223 62 Lund, Sweden

Document name
DOCTORAL THESIS

Date of disputation
2026-09-17

Sponsoring organization

Author(s)
Lucas Knuthson

Title and subtitle
Solving lattice protein problems with quantum computing

Abstract

Quantum computing has progressed from a theoretical concept to a practical experimental technology. By processing information according to quantum mechanics rather than classical rules, quantum computers may offer advantages for certain problems. Although a few algorithms have proven speedups, it remains unclear where quantum computers will be practically useful, especially since current devices are noisy, small, and not yet fault tolerant. This thesis explores quantum computing for biophysically motivated combinatorial problems, with a focus on lattice protein models. In these simplified models, a protein is represented as a chain on a discrete lattice. Despite their simplicity, lattice proteins retain important features of real proteins, such as such as the folding of a chain with exponentially many possible shapes to a unique minimum-energy structure and steric constraints.

To run lattice protein problems on quantum hardware, they must first be formulated using binary variables and objective functions. The latter typically contain soft constraints needed to ensure physical solutions. In Papers I–II, we developed and extended a coordinate-based mapping for lattice protein folding, enabling D-Wave's hybrid quantum-classical annealer to fold both 2D hydrophobic-polar and 3D Miyazawa-Jernigan lattice proteins. In Paper III, we adapted the approach to the gate-based paradigm and showed how quadratic soft constraints could be removed by a mixer developed for the maximum-independent-set problem. Papers IV–V addressed lattice protein design using quantum annealing, quantum approximate optimization algorithm, quantum alternating operator ansatz and hardware-efficient ansatz, while Paper VI extended the framework to multi-chain thermodynamics and the reconstruction of densities of states and heat-capacity features.

Together, these papers show how folding, design, and multi-chain thermodynamics can be formulated, implemented, and analyzed using present-day quantum and hybrid hardware. A recurring limitation is the treatment of soft constraints. In quantum annealing, their high connectivity leads to error accumulation. On gate-based hardware, they increase circuit depth and sensitivity to noise. At the same time, temporary constraint violations can provide useful freedom during optimization. Further progress will therefore require improved formulations, algorithms and hardware that reduce the cost of enforcing physical solutions while retaining this flexibility.

Key words
quantum annealing, quantum computing, statistical physics, optimization, Markov chains, Monte Carlo simulations, biophysics, lattice proteins

Classification system and/or index terms (if any)

Supplementary bibliographical information

Language
English

ISSN and key title

ISBN
978-91-90202-75-3 (print)
978-91-90202-76-0 (pdf)

Recipient's notes

Number of pages
168

Price

Security classification

Solving lattice protein problems with quantum computing

by Lucas Knuthson

LUND
UNIVERSITY

A doctoral thesis at a university in Sweden takes either the form of a single, cohesive research study (monograph) or a summary of research papers (compilation thesis), which the doctoral student has written alone or together with one or several other author(s).

In the latter case the thesis consists of two parts. An introductory text puts the research work into context and summarizes the main points of the papers. Then, the research publications themselves are reproduced, together with a description of the individual contributions of the authors. The research papers may either have been already published or are manuscripts at various stages (in press, submitted, or in draft).

Cover illustration front: Picture taken on Brunnhemsberget in Stenstorp Västra Götaland. Credit: Evrim Mete

Funding information: The thesis work was partly financially supported by the Swedish Research Council.

Faculty of Science, Computational Science for Health and Environment
Department of Earth and Environmental Sciences

isbn: 978-91-90202-75-3 (print)
isbn: 978-91-90202-76-0 (pdf)

Printed in Sweden by Media-Tryck, Lund University, Lund 2026

"'Zeno said this.' And what have you said?
'Cleanthes said that.' What have you said?

How much longer are you going to serve under others' orders?
Assume authority yourself and utter something
that may be handed down to posterity.
Produce something from your own resources.

This is why I look on people like this as a spiritless lot."

—Seneca, "Assume Authority Yourself," in *Why I Am a Stoic*, p. 48
translated by Robin Campbell

Contents

List of publications

This thesis is based on the following publications, referred to by their Roman numerals:

I **Folding lattice proteins with quantum annealing**

A. Irbäck **L. Knuthson**, S. Mohanty, C. Peterson
Phys. Rev. Res., 2022, 4, 043013

II **Folding lattice proteins confined on minimal grids using a quantum-inspired encoding**
A. Irbäck **L. Knuthson**, S. Mohanty
Phys. Rev. E. 2025 112, 045302

III **Penalty-free quantum optimization applied to lattice protein folding**

L. Gellersen **L. Knuthson**, A. Irbäck, S. Prestel
Submitted to. Phys. Rev. A

IV **Using quantum annealing to design lattice proteins**

A. Irbäck **L. Knuthson**, S. Mohanty, C. Peterson
Phys. Rev. Res., 2024, 6, 013162

V **Designing lattice proteins with variational quantum algorithms**

H. Linn **L. Knuthson**, A. Irbäck, S. Mohanty, L. García-Álvarez, G. Johansson
Accepted for publication in Phys. Rev. Appl.

VI **Thermodynamics of lattice protein systems using quantum annealing**

S. Grosinger, A. Irbäck, **L. Knuthson**, S. Mohanty, C. Peterson
Manuscript

All papers except Paper V use alphabetical author order.

List of abbreviations

HA Hybrid quantum-classical Annealing.

HEA Hardware-Efficient Ansatz.

HP Hydrophobic-Polar.

MJ Miyazawa-Jernigan.

QAOA Quantum Approximate Optimization Algorithm.

QAOA+ Quantum Alternating Operator Ansatz.

QPU Quantum Processing Unit.

QUBO Quadratic Unconstrained Binary Optimization.

Acknowledgements

During my PhD, I have been blessed with having a fantastic supervisor, Anders. From your ability to recognize which results are truly significant and maintain such a clear view of the bigger picture, to the way you meticulously read and writing scientific literature to, learning from you has been an immensely helpful. For this, I thank you deeply.

I would also be remiss to not thank my collaborators: Carsten for all the advice and fun stories; and Sandipan for all our discussion on programming. I also thank Stefan and Leif; your enthusiasm has been infectious. Furthermore, I wish to thank our collaborators in Gothenburg; Hanna, Laura and Göran, with a special thanks to Göran for encouraging me to be affiliated with WAQCT, which has brought me closer to the quantum technology research community in Sweden.

During this time I have also gotten to know some great people in COSHE. In, particular, thank you Anders Björkelund for all our fun discussions about football and other matters. Thank you Sebastian, for the year you have spent in our group.

I also want to thank the fellow PhDs, both past and present, at the department: Emil, for getting me into long-distance running; Thomas, for all the football and philosophical dicussions; Axel, for all the laughs and lunches; Daqu, for your continued encouragement; Suze for all the laughs. Furthermore, I would also like to thank Mikkel, Carlos, Alexander and Erik for making this journey more fun. A special thanks also go to Sebastian, Alexander and Thomas for providing feedback on this thesis.

During this journey I have been blessed with having great friends. While I am grateful to all of them, there are a few I would like to offer particular thanks to. Ashar, for our endless discussions about everything from fitness and popculture to philosophy and science; Frans, my Ultravasan running companion, for being the much-needed down-to-earth friend when the situation calls for it and for filling every room you enter with laughter and joy; Love, for being such a dependable and steadfast friend. Even things seem daunting, your support never falters; and lastly, Elias, for pretty much everything, but most importantly, for being an endless source of inspiration. It would be difficult to find better friends.

Lastly, I would like to thank the people who mean the most to me. First off, I thank my parents Per and Inger for listening to my excited and frustrated rants, and providing unconditional support. I thank my older brother, Alex for all the fun times and encouragement to keep going. Last but definitely not least, I would like thank my sunshine, Evrim for all your support and encouragement. You remain a constant source of inspiration and your presence makes my days much brighter.

Populärvetenskaplig sammanfattning på svenska

Kvantdatorer har fått mycket uppmärksamhet under de senaste åren på grund av sin möjliga förmåga att lösa vissa problem som är mycket svåra för vanliga datorer. De skiljer sig i grunden från dagens datorer genom att utnyttja kvantmekaniska fenomen som superposition, sammanflätning och tunnling. Superposition innebär förenklat att ett kvantsystem kan befinna sig i flera tillstånd samtidigt. Om uppgiften till exempel är att hitta vägen ut ur en labyrint, kan en klassisk dator behöva testa möjliga vägar en efter en. En kvantdator kan däremot, i princip, använda kvantmekaniska effekter för att behandla många möjliga vägar på samma gång.

Det finns redan några kända exempel där kvantdatorer teoretiskt ger tydliga fördelar. Till exempel kan den välkända Shors algoritm faktorisera stora tal snabbare än klassiska metoder. Samtidigt är det fortfarande oklart vilka praktiska problem kvantdatorer kommer att vara mest användbara för. En spännande utveckling är därför möjligheten att testa kvantalgoritmer på hårdvara för allt större system, även om hårdvaran ännu förblir brusig och därmed långt ifrån fullt felfri.

En viktig del av forskningen är därför att förstå hur dagens kvantdatorer faktiskt presterar på konkreta, svåra problem och förstå när, var och varför prestandan brister i praktiken när problemen blir större eller mer komplicerade. Ett intressant problemområde med breda tillämpningar är kombinatorisk optimering, där man letar efter den bästa lösningen bland ett mycket stort antal möjliga alternativ.

Min avhandling fokuserar på hur kvantberäkningsmetoder kan användas för att lösa biofysikaliskt motiverade optimeringsproblem. Fokus har främst legat på förenklade proteinmodeller, så kallade gitterproteiner. I en sådan modell beskrivs ett protein som ett pärlhalsband på ett rutnät, där varje pärla representerar en aminosyra. Modellen är betydligt enklare än ett verkligt protein, men fångar ändå centrala egenskaper, till exempel att många proteiner, trots ett astronmiskt antal möjliga strukturer, veckar sig till en specfik struktur.

I mina arbeten har jag behandlat tre problem för gitterproteiner. Det första är proteinveckning, där målet är att hitta den struktur som har lägst energi för en given proteinkedja. Det andra är det inversa problemet, det så kallade designproblemet, där man i stället söker en kedja som veckar sig till en viss önskad struktur. Det tredje är flerkedjeproblemet, där flera proteinkedjor tillåts växelverka med varandra.

En viktig del av mitt arbete har varit att utveckla lämpliga sätt att uttrycka dessa problem i binära 0-eller-1-variabler, vilket krävs för att kunna använda kvantdatorer. För att utvärdera algoritmerna har jag använt experiment på kvantdatorer och simulering-

ar av kvantsystemen på klassiska datorer.

Våra binära representationer av gitterproteiner använder strafftermer för att tillse att de slutliga lösningarna motsvarar korrekta kedjestrukturer. Under beräkningens gång kan brott mot detta villkor förekomma, vilket ibland kan underlätta sökandet efter den korrekta lösningen. Samtidigt leder dessa strafftermer till ökad bruskänslighet. I ett av arbetena visar vi teoretiskt hur användandet av strafftermer kan undvikas. Det återstår att se hur väl denna metod fungerar i praktiken. Fortsatta framsteg kräver troligtvis bättre algoritmer samt mindre brusig och mer flexibel hårdvara.

1 Introduction

Quantum computing has developed from a theoretical idea into an experimental reality. These computers utilize quantum phenomena such as quantum superposition, entanglement and tunneling. The phenomena will make fault-tolerant quantum computers able to perform certain computations at speeds believed to be unachievable by classical machines [1]. In recent years, an increasing number of quantum processors have become available, but they remain noisy, limited in size, and far from fault-tolerant.

The motivation for building quantum computers stems from a handful of algorithms with mathematically provable speedups over any known classical algorithm. These landmark algorithms include Shor's algorithm for integer factorization [2, 3] and Grover's algorithm for unstructured search [4], which provide an exponential and quadratic speedup, respectively. Similarly, quantum simulation is a natural application because quantum systems are often difficult to represent classically [1].

However, the number of problems for which quantum computers are known to provide clear advantages remains limited. Furthermore, these problems reveal little about the computers' utility for other tasks. Identifying new applications thus requires understanding how current quantum devices perform on concrete, challenging problems. Most such problems are not amenable to analytical solutions, especially for finite-sized systems in the presence of noise. Instead, this understanding comes from computational experiments, which involve gauging whether a problem is compatible with quantum hardware and where and why performance breaks down in practice.

Combinatorial optimization is a promising area for this search. Many computationally difficult scientific and engineering problems involve finding good or optimal solutions within a large discrete search space. By mapping the problems to quadratic functions of binary variables that can be addressed using quantum hardware, either analog quantum annealing or digital gate-based systems.

During my PhD, we have focused on exploring the use of quantum computers, both quantum annealers and gate-based systems, to help solve biophysically motivated combinatorial optimization problems. In particular, we investigated problems involving simplified lattice-based protein models (Fig. 1.1). In these models, a protein is represented as a self-avoiding chain on a lattice. Despite their simplicity, these models

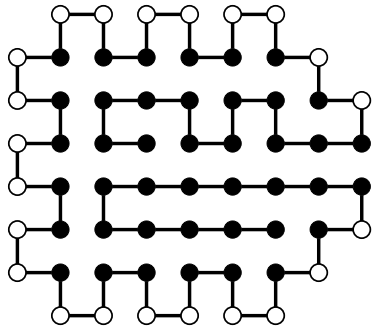


Figure 1.1: Example of a simplified lattice protein in two dimensions. The circles represent amino acids which, in this model, can only be hydrophobic (●) or polar (○). The energy function of this model favors a hydrophobic core.

can qualitatively capture essential properties of real proteins, such as the folding of a chain with exponentially many possible shapes to a unique minimum-energy structure. Lattice protein problems form a useful testing ground for quantum computing methods, since they have a simple and precise structure while remaining computationally challenging. Thus, they may reveal the strengths and limitations of emerging quantum algorithms. This thesis investigates how three central lattice protein problems, folding, design, and multi-chain thermodynamics, can be formulated and solved on present-day quantum hardware, and where the hardware falls short.

This thesis is split into two parts. The first part of the thesis, Ch. 1–Ch. 6, introduces and brings together three ingredients: combinatorial optimization, quantum computing and lattice protein models. In Ch. 2 I give a brief introduction to sampling and combinatorial optimization problems with classical algorithms. Chapter 3 introduces quantum computing, covering both quantum annealing and gate-based algorithms. Chapter 4 provides a brief description of lattice proteins and the folding, design and multi-chain thermodynamics problems. Chapter 5 shows how to transform these problems into binary forms that can be run on quantum hardware. The final chapter, Ch. 6, summarizes the papers and provides an outlook.

2 Classical computing

The quantum methods used in this thesis were evaluated against and partly inspired by classical methods. While Papers I–V concern optimization, Paper VI also addresses thermodynamics. In this section, I thus describe thermodynamic sampling and optimization. I begin with sampling, since the main classical optimization method used throughout the thesis, simulated annealing [5], builds directly on it.

In classical computing, the basic unit of information is the bit b, which can take one of two values, 0 or 1. A system of multiple bits is described by binary strings $\boldsymbol{b} = (b_0, .., b_{n-1}) \in \{0, 1\}^n$, where n is the number of bits. Such bitstrings can represent integers or encode the state of a computational process. The size of the state space grows exponentially with n, containing 2^n possible configurations.

Computation on bitstrings is performed using logical gates, which are deterministic operations mapping input bits to output bits. By composing such gates into circuits, arbitrary functions can be implemented, providing a universal model of classical computation.

2.1 Sampling

It is often desirable to compute thermal averages of observables at equilibrium. These averages are commonly estimated by sampling configurations $\boldsymbol{x} = (x_0, ..., x_{n-1})$ from the canonical Boltzmann distribution $p(\boldsymbol{x}) \propto \exp(-\beta E[\boldsymbol{x}])$, where $\beta = 1/T$ is the inverse temperature (I have set $k_B = 1$) and $E[\boldsymbol{x}]$ is the energy. By grouping configurations according to their energy $E[\boldsymbol{x}]$, one obtains the probability distribution

$$p(E) = e^{-\beta E} g(E) \Big/ \sum_{E'} e^{-\beta E'} g(E'), \tag{2.1}$$

where $g(E)$ is the density of states. A standard method for sampling from the Boltzmann distribution is by performing constant-temperature updates of the system which mimic thermal fluctuations. To do this, proposed updates are accepted according to the Metropolis criterion [6, 7], i.e. with probability

$$P_{\boldsymbol{x}\to\boldsymbol{x}'} = \min\left(1, e^{\beta(E[\boldsymbol{x}]-E[\boldsymbol{x}'])}\right), \tag{2.2}$$

where $\boldsymbol{x}$ and $\boldsymbol{x}'$ are the old and new configurations, respectively. Metropolis sampling is very time-consuming in rugged energy landscapes (see Fig. 2.1 for a schematic illustration of a rugged energy landscape) since it can become trapped in local minima separated by large energy barriers, so that $e^{\beta(E[\boldsymbol{x}]-E[\boldsymbol{x}'])}$ becomes very small.

An alternative to sampling from $p(\boldsymbol{x})$ is to estimate $g(E)$. Once obtained, $g(E)$ allows the computation of thermodynamic quantities at arbitrary temperatures. One way to estimate $g(E)$ is by using a weighted multi-histogram analysis [8–10]. We used this method in Paper VI. The idea is to draw random samples within an energy interval with equal probability. If enough samples are drawn, and the sampling is fair, this procedure will straightforwardly reconstruct the relative density of states within the energy interval. The procedure is repeated for many partially overlapping intervals. Each interval containing the energy level E' has its own estimate of $g(E')$. To combine the different estimates, each interval, j, that contribute to $g(E')$ is given a weight, $\alpha_j(E')$. The $\alpha_j(E')$'s are then iteratively updated to fulfill a set of self-consistency equations [8–10]. This yields $g(E)$ for all E up to a multiplicative constant. To have a unique solution, it is common to impose $\sum_E g(E) = 1$.

To provide benchmarks of the thermodynamic quantities obtained in Paper VI, we used the Wang-Landau algorithm [11]. This algorithm starts from an initial estimate $g'(E)$ and performs a random walk in configuration space. A proposed update from $\boldsymbol{x}$ to $\boldsymbol{x}'$ is accepted with probability

$$P_{\boldsymbol{x}\to\boldsymbol{x}'} = \min\left(1, g'(E[\boldsymbol{x}])/g'(E[\boldsymbol{x}'])\right). \tag{2.3}$$

Each visit to an energy level updates $g'(E)$ by multiplicative modification factor, and a corresponding entry of an energy histogram is incremented. When the histogram is sufficiently flat, the modification factor is reduced, the histogram is reset and the procedure is repeated. This yields a progressively refined estimate of the true density of states $g(E)$ [11]. The moves serves to ensure the random walk is biased towards undervisited energy levels.

2.2 Optimization

In optimization, one seeks variables $\boldsymbol{x}$ that minimize (or maximize) a function $f(\boldsymbol{x})$. For general functions, this is often a difficult task. Many challenging problems, such as the traveling salesman and scheduling problems [12], fall into this category. Minimizing a function naturally connects to energy minimization in thermodynamics, and optimization can be seen as sampling at zero temperature. This thesis is primarily concerned with solving lattice protein optimization problems. These problems are discrete, and are therefore often referred to as combinatorial optimization problems.

Combinatorial optimization

Combinatorial optimization problems involve finding an optimal solution, $\boldsymbol{x}^*$, from a finite set of discrete configurations, i.e.

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}). \tag{2.4}$$

Here $\mathcal{X}$ is typically a structured discrete set, such as $\{0,1\}^n$, permutations, graph partitions or protein-chain configurations. Constraints are often incorporated into the minimization problem by restricting the state space $\mathcal{X}$ or via penalty terms. The latter yields an unconstrained formulation, where infeasible states can be visited, albeit with a suboptimal value of f.

Many combinatorial optimization problems can be expressed as Quadratic Unconstrained Binary Optimization (QUBO) problems [12]. Here one seeks a $\boldsymbol{b}^*$ such that

$$\boldsymbol{b}^* = \arg\min_{\boldsymbol{b}\in\{0,1\}^n} E(\boldsymbol{b}), \quad E(\boldsymbol{b}) = \sum_{i=0}^{n-1} A_i b_i + \sum_{i<j}^{n-1} B_{ij} b_i b_j = \sum_{i\leq j}^{n-1} b_i Q_{ij} b_j, \tag{2.5}$$

with $b_i \in \{0,1\}$. Note that the diagonal items in the last sum are linear since $b_i^2 = b_i$. To recast a problem into a QUBO form, one thus need to choose $Q_{i,j}$ such that $\boldsymbol{b}^*$ represents the solution to the desired problem. All QUBO problems can be identified with a graph where the linear terms, $A_i b_i$, are represented as nodes and the quadratic terms, $B_{ij} b_i b_j$, as edges.

Alternatively, one may consider spin variables $s_i \in \{-1,1\}$ — which represent spin-↓ and spin-↑, respectively— and minimize the Ising spin-glass Hamiltonian [12]

$$E(\boldsymbol{s}) = \sum_{i=0}^{n-1} h_i s_i + \sum_{i<j}^{n-1} J_{ij} s_i s_j, \quad \boldsymbol{s} = (s_0, .., s_{n-1}). \tag{2.6}$$

The two formulations are equivalent under the transformation $b_i = (1 - s_i)/2$. This mapping naturally connects combinatorial optimization to statistical physics, where such energy functions describe interacting spin systems.

Simulated annealing

A standard method for energy minimization, which we used as a classical benchmark in Papers I and II, is simulated annealing [5]. Here, one performs configurational updates at decreasing temperatures (β increases). The proposed updates are often accepted according to the Metropolis criterion [Eq. (2.2)]. The algorithm mimics thermal fluctuations and has a nonzero probability of accepting moves that increase

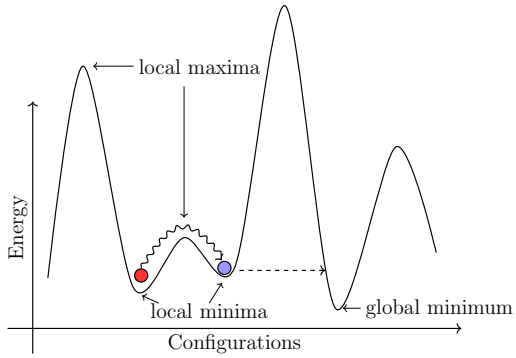


Figure 2.1: Illustration of a rugged energy landscape with local maxima and minima. The Metropolis algorithm (represented by the red ball), and thus in extension also simulated annealing, traverse this landscape using classical thermal fluctuations. Since thermal fluctuations can increase the energy, a sufficiently large fluctuation (the squiggly line) can make the system "jump over" energy barriers between minima. In comparison, quantum methods (represented by the blue ball) described in Ch. 3 traverses the landscape using quantum dynamics. According to the standard picture, this allows the quantum system to "tunnel" through high-energy barriers. This forms the basic intuition for the algorithms described in Ch. 2 and 3.

the energy. It can thus escape local minima, especially when β is small. Since these fluctuations are gradually reduced when decreasing the temperature, the dynamics becomes increasingly biased toward low-energy configurations. For infinite spin systems [Eq. (2.6)], increasing β may also take the system through a thermal phase transition. In spin-glass problems, the low-temperature phase is characterized by a rugged energy landscape and slow dynamics.

Simulated annealing is guaranteed to find the ground state of an optimization problem given sufficiently long computational time and suitable ergodicity [13, 14]. However, if the cooling is too fast for the system, it may freeze into a local minimum instead of reaching the ground state. For some problems it can thus take a very long time to find the minimum. Such optimization problems are often characterized by a rugged energy landscape (see Fig. 2.1) as well.

3 Quantum computing

There are two main tracks to solve the optimization problems of Ch. 2 on quantum computers, quantum annealing and gate-based quantum algorithms. This chapter presents both approaches along with some technical details of the two different types of quantum computers. Throughout the thesis, I will assume familiarity with the basic formalism of nonrelativistic quantum mechanics. This includes the Dirac notation and the representation of quantum states as complex-valued vectors in a Hilbert space. I will also assume knowledge of the Schrödinger equation and the probabilistic interpretation of measurement outcomes. I will set $\hbar = 1$.

3.1 Quantum computing basics: Qubits

Before turning to quantum optimization algorithms, I will introduce the unfamiliar reader to the qubit. Experts can thus skip this section and directly move to Sec. 3.2. In quantum computations, information is represented and processed using quantum bits, or qubits [1]. In contrast to classical binary bits, qubits may exploit intrinsically quantum-mechanical effects and can be regarded as a generalization of the classical bit. A single qubit $|\psi\rangle$ in the Hilbert space $\mathbb{H}$ is commonly written in terms of two complex amplitudes ψ_0, ψ_1 as

$$|\psi\rangle = \psi_0 |0\rangle + \psi_1 |1\rangle = \begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix}, \tag{3.1}$$

where the state is normalized according to $\langle\psi|\psi\rangle = |\psi_0|^2 + |\psi_1|^2 = 1$ [1, 15, 16]. The final equality in Eq. (3.1) holds specifically in the $|0\rangle, |1\rangle$ basis, where the two basis states are represented by the standard basis vectors [1, 15]

$$|0\rangle = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{3.2}$$

In practice, quantum computations are carried out on systems containing several qubits.

As in the single-qubit case, the state of a multi-qubit system can be expressed through a set of complex expansion coefficients. For a register of n qubits, however, the number of coefficients increases from two to 2^n. Thus, the state $|\psi\rangle$ may be written in terms of the amplitudes $\psi_0, \ldots, \psi_{2^n-1}$ as [1, 15]

$$|\psi\rangle = \psi_0 |0\cdots00\rangle + \psi_1 |0\cdots01\rangle + \ldots + \psi_{2^n-1} |1\cdots11\rangle = \begin{pmatrix} \psi_0 \\ \psi_1 \\ \vdots \\ \psi_{2^n-1} \end{pmatrix}. \tag{3.3}$$

The basis vectors appearing in Eq. (3.3) are constructed as $|q_0 q_1 \cdots q_{n-1}\rangle = |q_0\rangle \otimes |q_1\rangle \otimes \ldots \otimes |q_{n-1}\rangle$, where $q_i = 0, 1$ denotes the state of the i-th qubit. This basis is referred to as the computational basis. The index j on ψ_j is the integer representation of the binary state in its corresponding ket, i.e. $j = \sum_i^{n-1} q_i 2^i$.

3.2 Quantum annealing

The idea of quantum annealers is to exploit the adiabatic theorem in quantum mechanics to find the ground state of a Hamiltonian H_C [1]. The theorem considers a time-dependent Hamiltonian of the form [15–20]

$$H(t) = A(t)H_M + B(t)H_C, \quad t \in [0, \tau]. \tag{3.4}$$

Here, the coefficient multiplying H_C is varied from zero to one, while that multiplying H_M is varied from one to zero. Hence, the Hamiltonian satisfies $H(0) = H_M$ and $H(\tau) = H_C$. The theorem states that if the system is in the nondegenerate ground state of $H(t)$ at $t = 0$ and the Hamiltonian changes sufficiently slowly, the system will remain in the instantaneous ground state after a total annealing time τ. For this to hold, the annealing time must be chosen such that [17, 20]

$$\tau \gg \max_{0\le s\le 1} \frac{|\langle\psi_e(s)| \frac{dH(s)}{ds} |\psi_g(s)\rangle|}{(E_g(s) - E_e(s))^2}, \quad s \equiv \frac{t}{\tau} \tag{3.5}$$

where $|\psi_g(s)\rangle$ and $|\psi_e(s)\rangle$ denote the instantaneous ground and first excited states, respectively, with corresponding energies $E_g(s)$ and $E_e(s)$. Equation (3.5) also shows that the Hamiltonians H_M and H_C in Eq. (3.4) must be chosen such that they do not commute. If they did commute, there would exist a point s_1 at which the gap $|E_g(s_1) - E_e(s_1)|$ is identically 0 [16, 20].

[1] Quantum annealing is deeply tied to adiabatic quantum computation. I will, for the sake of simplicity, refer to both as quantum annealing.

To use quantum annealing for optimization, the initial Hamiltonian in the adiabatic theorem, H_M [2], is selected such that its ground state is simple to prepare. In current implementations, the initial Hamiltonian is chosen as [15, 16]

$$H_M = -\frac{1}{n}\sum_{i=0}^{n-1}\sigma_i^x = H_X, \tag{3.6}$$

for a system containing n qubits. Here, $\sigma_i^x = I \otimes ... \otimes \sigma^x \otimes ... \otimes I$, where I is the identity matrix and the Pauli matrix σ^x acts on the i-th qubit. In the σ^z basis, this choice prepares the system initially in the state [16]

$$|\psi(0)\rangle = \left(\frac{1}{\sqrt{2}}\right)^n (|0_0\rangle + |1_0\rangle) \otimes \cdots \otimes (|0_{n-1}\rangle + |1_{n-1}\rangle) = |+\rangle \tag{3.7}$$

where $|0\rangle$ and $|1\rangle$ are interpreted as the spin-↑ and spin-↓ states, respectively. Thus, the initial state is a uniform superposition of all computational basis states.

The final Hamiltonian, H_C, is chosen as a quantum analog of Eq. (2.6) [and, by extenstion, Eq. (2.5)]. It is obtained by expressing the classical energy functions in terms of quantum operators, i.e. by replacing s_i with σ_i^z in Eq. (2.6). The final Hamiltonian is thus chosen as [15, 16, 19]

$$H_C = \sum_{i=0}^{n-1} h_i\sigma_i^z + \sum_{i<j}^{n-1} J_{ij}\sigma_i^z\sigma_j^z, \tag{3.8}$$

whose ground state, $|q_0...q_{n-1}\rangle$, encodes the solution of the problem under consideration. Note that to use quantum annealing machines H_C thus needs to be quadratic. Since the anneal begins in the the ground state of H_X [Eq. (3.7)], the system may exploit quantum tunneling during the evolution in order to reach the ground state of the Hamiltonian in Eq. (3.8) (see Fig. 2.1). Quantum annealing can thus be seen as a quantum analog of simulated annealing, where the thermal fluctuations are replaced by quantum ones with strength governed by $A(t)/B(t)$.

Analogously to simulated annealing, for infinite systems quantum annealing typically crosses a quantum phase transition between the initial transverse-field dominated phase and the ordered phase of the final problem Hamiltonian. The finite-size scaling of the minimum energy gap, and thus the algorithmic performance, depends on the nature of this transition. First-order transitions often exhibit gaps that shrink exponentially with n, whereas continuous transitions have polynomial gap scalings [21, 22]. Mean-field models (with fully-connected H_Cs) of some constraint-satisfaction problems possess a first-order transition [22]. In contrast, recent results show that the

[2] Throughout this chapter, I will use H_M to refer to a general mixer (called Driver in quantum annealing contexts) and H_X to the specific one presented in Eq. (3.6).

energy gap scales polynomially for two- and three-dimensional Ising spin glasses with nearest-neighbor interactions and random couplings $J_{i,j} = \pm 1$ [23–25]. However, the ordered phase of these spin-glass Hamiltonians has not yet been explored and may still present challenges for the algorithm, as it does for simulated annealing.

Even if quantum annealing does not provide a general advantage for finding exact ground states, it may still be useful as a heuristic optimization method. In particular, several studies indicate that quantum annealers can sometimes be faster than classical methods at finding good low-energy approximate solutions [26, 27].

D-Wave quantum annealers

D-Wave Quantum is a company that produces commercially available heuristic solvers based on quantum annealing. In this subsection, the operating principles of these devices are briefly summarized. For a more detailed account of the underlying physics, the reader is referred to Refs. [28–30]. The qubits used in D-Wave machines are flux qubits implemented using superconducting circuits. These flux qubits are connected through couplers, and both the qubits and couplers are controlled using external magnetic fields. The coefficients h_i and J_{ij} in the final Hamiltonian in Eq. (3.8) are set by time-independent magnetic fields applied to the qubits and couplers, respectively. The annealing schedule itself is controlled by time-dependent magnetic fields, which transform the Hamiltonian in Eq. (3.6) into the final Hamiltonian in Eq. (3.8) [15]. The complete network of qubits and couplers is referred to as the quantum processing unit (QPU).

The QPU is affected by several limitations that can influence the outcome of the annealing process. Some of these originate from the environment, such as the operating temperature, which affects coherence, or high-energy photons that may induce unwanted qubit excitations [31, 32]. Other limitations are intrinsic to the control of the device and are known as integrated control errors [31, 32]. These errors arise because the Hamiltonian in Eq. (3.8) is imperfectly implemented, resulting in an effective Hamiltonian of the form [31–35]

$$H_\delta = \sum_{i=0}^{n-1} (h_i + \delta h_i)\sigma_i^z + \sum_{i<j}^{n-1} (J_{ij} + \delta J_{ij})\sigma_i^z \sigma_j^z, \tag{3.9}$$

where δh_i and δJ_{ij} represent random noise terms. D-Wave annealers often rescale the given Hamiltonian with the largest h_i or $J_{i,j}$ [36], and thus the error terms can have a comparable size to the user-given terms. In a slow enough anneal, the D-Wave device therefore finds the ground state of Eq. (3.9) instead of Eq. (3.8) [33]. It has been shown that, given a realistic strength of the noise, the probability that these

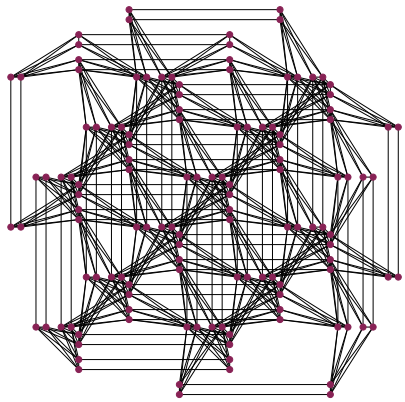


Figure 3.1: Pegasus graph of size 3 (P_3) where the nodes are shown as burgundy dots and the edges as gray lines.

two Hamiltonians share the same ground state decreases exponentially as the number of qubits n increases [33, 37]. As a consequence, the control errors may lead to an exponentially decaying success rate.

While noise is often seen as an obstacle, the environmental coupling of the QPU can be useful. Since the device operates at finite temperature and interacts with its surroundings during the anneal, the samples generated by the QPU may, in certain parameter regimes, approximate a thermal distribution of the programmed Ising Hamiltonian with an effective temperature. This observation has motivated the use of D-Wave annealers as heuristic samplers in statistical physics[38–40] and machine-learning contexts [41, 42].

In addition to noise, D-Wave devices are also constrained by the sparse connectivity of the QPU. The QPU primarily used in this thesis, Advantage, contains more than 5000 qubits, with each qubit connected to at most 15 couplers. Its topology is the Pegasus graph of size 16, denoted P_{16} [15, 43] (see Fig. 3.1 for a P_3 graph). Because the Pegasus graph is sparsely connected, not every problem graph can be mapped directly onto a subgraph of the hardware graph. To address this, D-Wave uses minor embedding, in which logical variables are represented by chains of physical qubits. The qubits within such a chain are coupled so strongly that they behave effectively as a single qubit (see Fig. 3.2). To keep a chain intact, the strength of the chain interactions must typically be stronger than the interaction between logical variables [44]. The known upper bound on the number of additional qubits required by this procedure scales quadratically with the problem size [45]. These logical chain qubits can have greater connectivity than the underlying physical qubits, allowing a larger class of problems to be solved, although at the cost of using more qubits [44].

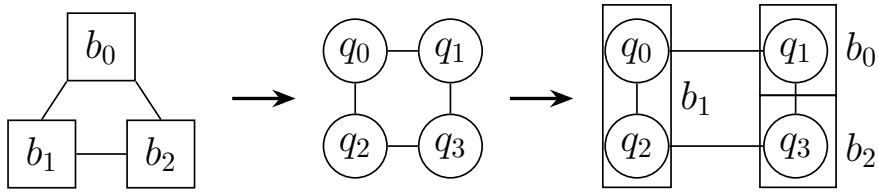


Figure 3.2: Representation of how an embedding is made on the D-Wave machines. A triangular graph with nodes b_0, b_1 and b_2 is embedded in a square graph, with nodes q_0, q_1, q_2 and q_3, by letting q_0 and q_2 represent the logical variable b_1. In this case, q_0 and q_2 form a chain of length 2. The triangular graph is the problem graph and the graph furthest to the right represents its embedding into the QPU topology.

Highly connected QUBO problems, such as the penalty terms in Ch. 5, require longer chains and correspondingly stronger chain interactions. The susceptibility to integrated control errors [Eq. (3.9)] increases with chain size since the user-given problem is rescaled with the largest chain interaction [36]. The embedded problem thus becomes increasingly sensitive to noise.

Hybrid quantum-classical annealers (HA)

Besides solvers that rely solely on quantum annealing, D-Wave also provides hybrid quantum-classical solvers (HA), which were used in Papers I, II, IV and VI. The detailed internal operation of these hybrid solvers is proprietary information; see the documentation for Leap Hybrid solvers [46]. At a high level, the solver operates by combining classical solvers with calls to the QPU for selected subproblems of the user-given problem. The subproblems are generated by classical heuristic partitioning methods and are chosen to be small enough to be handled efficiently by the QPU. The solutions returned from the QPU are then used to guide the classical optimization process [46] (see Fig. 3.3). The purpose of this approach is to accelerate the solution of difficult optimization problems by incorporating QPU queries into a classical search procedure. The more recent hybrid solvers make it possible to address problems with many tens of thousands of fully-connected variables, which is far beyond what can be studied with the QPU alone.

3.3 Gate-based quantum computing

While quantum annealing machines can address optimization problems, other applications often require greater flexibility than these machines can offer. Experimental

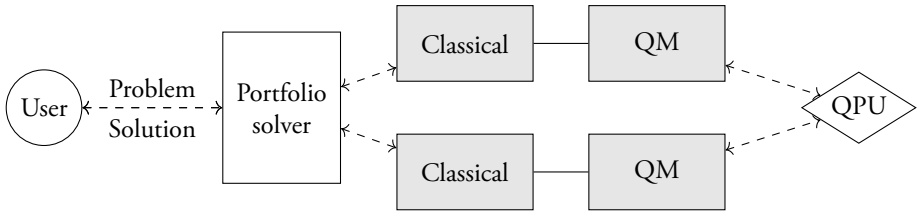


Figure 3.3: Schematic illustration of the hybrid quantum-classical solvers. The portfolio solver reads the input from the user, starts several hybrid solvers and calculates a time limit (or uses the one given by the user). The hybrid solvers contain a classical algorithm and a quantum module (QM). The classical solvers attempt to solve the problem, while the QM partitions the problem into subproblems and sends them to the QPU. The QPU responds with the solutions to the subproblems, and these responses help to guide the classical solver. At the time limit, the hybrid solvers deliver their results to the portfolio, which picks the best solution and sends it to the user.

efforts have therefore increasingly focused on gate-based quantum computers. This motivated us to investigate gate-based quantum algorithms in Papers III and V. In contrast to analog quantum annealing, which follows the time evolution of a system, here a discrete set quantum gates are applied onto a state to perform computations. The gates are quantum analogs of classical logical gates and are represented as matrices acting on a vector, using the computational (σ^z) basis. For example, a gate A acting on the state in Eq. (3.2) can be represented as [1, 16]

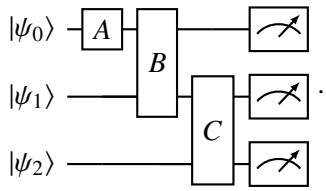
$$\begin{aligned} A|\psi\rangle &= A\left(\psi_0|0\rangle + \psi_1|1\rangle\right) \\ &= \begin{pmatrix} a & b \\ c & d \end{pmatrix}\begin{pmatrix} \psi_0 \\ \psi_1 \end{pmatrix} = (\psi_0 a + \psi_1 b)|0\rangle + (c\psi_0 + d\psi_1)|1\rangle \end{aligned} \tag{3.10}$$

in the $|0\rangle, |1\rangle$ basis. Conservation of probability requires the norm of $A|\psi\rangle$ to be 1 for general $|\psi\rangle$'s and thus all gates must be unitary, i.e. $AA^\dagger = I$.

In a multi-qubit system of n qubits, a gate may act on a single or several qubits. A gate A_i acting only on the i-th qubit is extended to the full n-qubit system as $A_i = I_0 \otimes \cdots I_{i-1} \otimes A_i \otimes I_{i+1} \otimes \cdots \otimes I_{n-1}$, leaving all other qubits unaffected, just as in Eqs. (3.6) and (3.8). Gates acting on several qubits simultaneously, e.g. A_{ijk}, are extended in the same way. In principle, a gate may act on an arbitrary number of qubits. In practice, however, directly implementing many-qubit gates is technically challenging, and quantum computations are thus typically constructed from one- and two-qubit gates [1]. The complexity of a gate-based algorithm is thus partly determined by the number and arrangement of the required two-qubit gates, which becomes important for highly connected optimization problems.

The sequence of gates is called a quantum circuit and is often represented in a di-

agram. Consider a system of three qubits initially in the state $|\psi_0, \psi_1, \psi_2\rangle$. One acts with the sequence of gates, A_0, $B_{0,1}$, and $C_{1,2}$ such that the resulting state is $C_{1,2}B_{0,1}A_0\,|\psi_0, \psi_1, \psi_2\rangle$, which is subsequently measured. The diagram of this circuit is [1, 16]



(3.11)

Here, the qubits are represented as wires, gates as boxes and measurements as boxes with arrows at the end of the wires. The circuit is read left to right (note the reverse order of the gates compared to the mathematical expression of the circuit). The computational cost is often measured in the circuit depth (the longest sequence of gates in the computation that can not be executed in parallel). All gate-based quantum algorithms, including Shor's algorithm for prime factorization [2, 3] and Grover's unstructured search algorithm [4], can be represented by quantum circuit diagrams [1, 16].

Some common quantum gates include single-qubit rotation gates and two-qubit entangling gates. Single-qubit gates are often expressed as rotations using the Pauli matrices [1, 16]

$$e^{-i\theta\sigma^k/2} = R_k(\theta) = \quad -\boxed{R_k(\theta)}- \,, \tag{3.12}$$

where σ^k is the k-th Pauli matrix. An important special case is the Hadamard gate [1, 16]

$$-\boxed{H}- \;= \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \tag{3.13}$$

which creates a superposition in the computational basis by turning $|0\rangle$ into $|+\rangle$ [Eq. (3.7)]. Two-qubit gates introduce entanglement and the most common example is the controlled-NOT (CNOT) gate, which flips a target qubit, $|\psi_1\rangle$, conditioned on whether the control qubit, $|\psi_0\rangle$, is in state $|1\rangle$, i.e. [1, 16]

$$\text{CNOT} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} = \quad . \tag{3.14}$$

Since it is a commonly used gate, it has a special circuit representation shown on the right in Eq. (3.14). In practice, different quantum computers often implement hardware specific entangling gates together with arbitrary single-qubit rotations, from which standard gates like CNOT are constructed [16].

A gate set is said to be universal if any unitary operation on an arbitrary number of qubits can be approximated to arbitrary accuracy using only gates from that set. An important example of a universal gate set is the set consisting of all single-qubit rotations together with the CNOT gate [1, 16].

Quantum approximate optimization algorithm (QAOA)

The most famous algorithm to solve combinatorial optimization problems using gate-based quantum computers is the Quantum Approximate Optimization Algorithm (QAOA) [47], which we used as a baseline in Papers III and V. It is inspired by quantum annealing [16, 47] and is, to my knowledge, one of the first examples of a variational quantum algorithm. Consider the Hamiltonians in Eqs. (3.4), (3.6) and (3.8) and an initial state $|\psi(t=0)\rangle = |+\rangle$. After a time τ, the state will have evolved to

$$|\psi(t)\rangle = U(\tau)\,|\psi(0)\rangle = \mathcal{T}\exp\left[-i\int_0^\tau dt H(t)\right]|+\rangle\,, \tag{3.15}$$

where $\mathcal{T}$ is the time-ordering operator. Using p equally sized time intervals $\Delta t = \tau/p$, the evolution operator $U(\tau)$ can be expanded to first order as [16, 47]

$$\begin{aligned} U(t) &\approx \prod_{k=1}^{p} \exp[-iH(t_k)\Delta t] \\ &\approx \prod_{k=1}^{p}\Big[\exp[-iH_X A(t_k)\Delta t]\exp[-iH_C B(t_k)\Delta t] + O\Big((\Delta t)^2\Big)\Big] \end{aligned} \tag{3.16}$$

which is exact in the limit $\Delta t \to 0$. The $2p$ exponentials $\exp[-iH_X A(t_k)\Delta t]$ and $\exp[-iH_C B(t_k)\Delta t]$[3] can be interpreted as collections of quantum gates, where each term in Eqs. (3.6) and (3.8) gives rise to a one- or two-qubit rotational gate acting on $|+\rangle$. The resulting circuit approximates $|\psi(t)\rangle$ and can become very deep for highly connected H_Cs. A large p is needed to ensure that the circuit approximates $|\psi(t)\rangle$ well [16, 47].

Each exponential in the product in Eq. (3.16) has a unique constant, either $A(t_k)\Delta t$ or $B(t_k)\Delta t$, multiplied with H_X or H_C. Renaming these constants as

$$\boldsymbol{\beta} = (\beta_1, ..., \beta_p) = \Big(A(t_1)\Delta t, ..., A(t_p)\Delta t\Big) \tag{3.17}$$

and

$$\boldsymbol{\gamma} = (\gamma_1, ..., \gamma_p) = \Big(B(t_1)\Delta t, ..., B(t_p)\Delta t\Big)\,, \tag{3.18}$$

[3] While QAOA can in principle handle higher than quadratic H_Cs, in practice the many-qubit gates need to be decomposed into two-qubit gates, thereby increasing the circuit depth.

a state can be defined as

$$|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle = \prod_{k=1}^{p} e^{-iH_X\beta_k} e^{-iH_C\gamma_k} |+\rangle \,. \tag{3.19}$$

Here a circuit consisting of alternating layers of the rotational gates $\exp[-iH_X\beta_k]$ and $\exp[-iH_C\gamma_k]$ generates $|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle$ by acting on $|+\rangle$, which can be created by applying Hadamard gates on $|0\rangle$. The conceptual leap of QAOA is to regard $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ as variational parameters that can be optimized classically, rather than as parameters specifying a predetermined time evolution [16, 47].

This variational perspective allows the expectation value of H_C in the state $|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle$ to be interpreted as a cost function,

$$F(\boldsymbol{\gamma},\boldsymbol{\beta}) = \langle \boldsymbol{\gamma},\boldsymbol{\beta}|H_C|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle = \sum_z \text{Prob}(z \mid \boldsymbol{\gamma},\boldsymbol{\beta})C(z), \tag{3.20}$$

parameterized by $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$. $C(z)$ is the classical cost and $\text{Prob}(z \mid \boldsymbol{\gamma},\boldsymbol{\beta})$ is the probability of the bitstring z, given the parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$. Successful optimization implies $\text{Prob}(z^* \mid \boldsymbol{\gamma},\boldsymbol{\beta}) \gg 0$ for the ground state, z^*[4]. To give ourselves the best chance of achieving this, $F(\boldsymbol{\gamma},\boldsymbol{\beta})$ is optimized classically over the variational parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$. In doing so, hopefully the circuit uses a smaller p than what is needed to approximate the time evolution in Eq. (3.15). In terms of annealing, this process can be seen as discretizing $A(t)$ and $B(t)$ and subsequently optimizing the shape of the two functions [16, 47].

Identifying the best variational parameters is itself a difficult optimization task. Indeed, it relies on optimizing a function of $2p$ variables where p should be sufficiently large to ensure success. Thus it is common to start the optimization from predetermined good parameters, which are often taken from similar and already optimized problems [48, 49].

All in all, the QAOA algorithm can be summarized as follows. Initialize the qubits in $|+\rangle$ using Hadamard gates. Apply p alternating layers of a mixer operator, $\exp[-iH_X\beta_k]$, and a phase separator, $\exp[-iH_C\gamma_k]$. For many optimization problems, circuit can become rather deep since H_C may be densely connected and requires on the order of $O(n^2)$ gates per layer for n-qubits. After applying these layers to the qubits, measurements are performed. This process is repeated multiple times to estimate the objective function $F(\boldsymbol{\gamma},\boldsymbol{\beta})$. The variational parameters are then updated, and the circuit is executed again. This is iterated until a stopping criterion is reached [16, 47]. A schematic illustration of the algorithm can be found in Fig. 3.4.

[4]Strictly, minimizing $F(\boldsymbol{\gamma},\boldsymbol{\beta})$ does not guarantee that the ground-state probability is maximized.

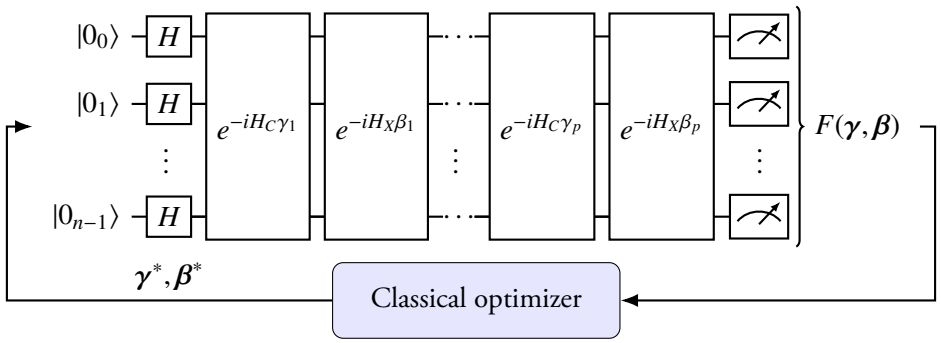


Figure 3.4: Outline of the QAOA algorithm. Classical parameters $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are fed into a quantum circuit consisting of one layer of Hadamard gates and p layers alternating exponentials [see Eq. (3.16)]. The measured states are used to estimate $\langle\boldsymbol{\gamma},\boldsymbol{\beta}|H_C|\boldsymbol{\gamma},\boldsymbol{\beta}\rangle = F(\boldsymbol{\gamma},\boldsymbol{\beta})$. The parameters are optimized classically to minimize $F(\boldsymbol{\gamma},\boldsymbol{\beta})$.

Quantum alternating operator ansatz (QAOA+)

Let us discuss the role of the mixer operator in QAOA. Its purpose is to generate transitions between computational basis states. The mixer Hamiltonian H_X induces single-bit flips, connecting computational basis states that differ by a single bit value. Through successive applications, this enables transitions between any pair of basis states. However, in many problems only a subset of the basis states represent feasible solutions. With H_X, and thus in standard QAOA, this is solved by turning these hard constraints into soft constraints by adding penalty terms to H_C, which penalize infeasible states. This can make H_C highly connected even when the desired energy function is sparse.

An alternative to penalizing infeasible states with soft constraints is to instead restrict the dynamics of the algorithm to the feasible subspace. Indeed, the replacement of $\exp[-iH_X\beta_k]$ with a parameterized constraint preserving mixer circuit $U_M(\beta_k)$ is the central modification introduced in the Quantum Alternating Operator Ansatz (QAOA+) [50]. Instead of encoding hard constraints as soft penalties in the objective function, $U_M(\beta_k)$ is chosen such that the dynamics remain entirely within the feasible subspace, which may move the complexity from H_C to $U_M(\beta_k)$.

While the mixer unitary in QAOA+ can take any form, the simplest constructions are achieved by exponentiating a single mixer Hamiltonian, $U_M(\beta_k) = \exp[-i\beta_k H_M]$. A prominent example is the XY-mixer [50–53],

$$H_{XY} = -\sum_{(i,j)\in E_M}\left(\sigma_i^x\sigma_j^x + \sigma_i^y\sigma_j^y\right), \tag{3.21}$$

where E_M describes which qubits are connected by the mixer. This mixer Hamiltonian generates transitions only between computational basis states with equal Hamming weights[5], and therefore removes the need for soft constraints meant to enforce a fixed number of qubits in $|1\rangle$, such as one-hot encodings. In Papers III and V, we mainly used XY-mixers with all-to-all connections [51, 52].

The use of constraint-preserving mixer circuits introduces an additional practical complication. Since $U_M(\beta_k)$ no longer connects the full computational basis, the initial state must already lie in the desired feasible subspace. In the case of H_{XY} this can be done by creating so-called Dicke states, which can be generated efficiently [55]. For general mixer circuits, however, preparing such a state can be substantially more difficult than for H_{XY} or H_X. In some applications, one therefore initializes the algorithm in a classically obtained feasible bitstring rather than in a superposition over feasible states.

Apart from this modified choice of mixer circuit and initial state, the overall algorithmic structure of QAOA+ remains the same as in standard QAOA. Both algorithms consist of p alternating applications of $U_M(\beta_k)$ and $\exp[-i\gamma_k H_C]$, and a subsequent classical optimization of $F(\boldsymbol{\gamma},\boldsymbol{\beta})$ over $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ [50], which makes both variational algorithms. This optimization can again be challenging, but parameter-initialization schemes similar to those used in QAOA may facilitate it. The algorithmic structure is thus similar to that of Fig. 3.4, but with the Hadamard gates replaced by the initialization gates and $\exp[-iH_X\beta_k]$ replaced by $U_M(\beta_k)$.

In Paper III, we proposed and explored a QAOA+ algorithm for lattice protein folding (see Ch. 4) based on the QUBO formulation introduced in Paper I. In the latter, all soft constraints have closed-form quadratic expressions. Every penalty term can therefore be mapped on to an edge in a conflict graph, with the qubits as nodes. For a valid chain structure, any pair of neighbors in this graph can not be both in the state $|1\rangle$. The quantum evolution of the system can be resticted to the space of states fulfilling this condition by using a mixer circuit that changes a site from $|0\rangle$ to $|1\rangle$ only if none of its neighbors are in $|1\rangle$. A mixer that creates such dynamics can be defined as [52, 56, 57]

$$U_M(\beta_k) = \prod_i V_i(\beta_k) \qquad V_i(\beta_k) = e^{-i\beta_k \sigma_i^x \bar{B}_i} \qquad \bar{B}_i = \prod_{j\in N_i} \frac{1+\sigma_j^z}{2}, \tag{3.22}$$

where N_i is the set of neighbors to qubit i in the constraint graph. V_i can be expressed as

$$V_i(\beta) = I + (e^{-i\beta\sigma_i^x} - I)\bar{B}_i. \tag{3.23}$$

Hence, V_i amounts to σ_i^x rotation of qubit i if its neighboring nodes are inactive (0), which is checked by $\bar{B}_i$. Whenever any of its neighbors is in the active $|1\rangle$ state, qubit i

[5] This gives a quantum analog of Kawasaki dynamics [54] in the classical Ising model.

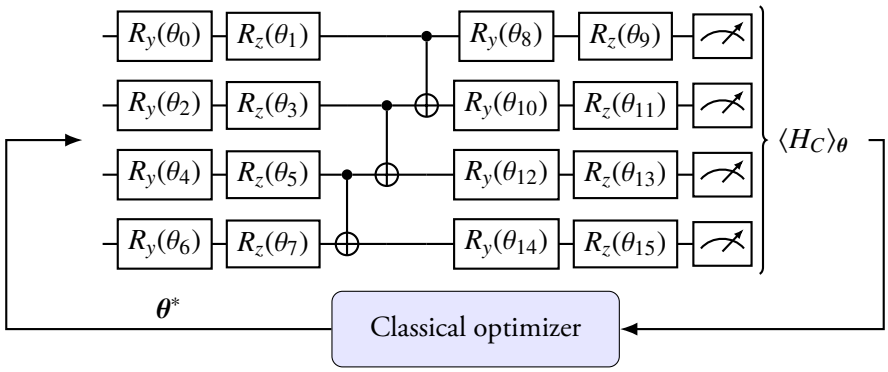


Figure 3.5: Outline of the HEA approach, using a single-layer hardware-efficient quantum circuit [58] and four qubits. An entangling layer of CNOT gates in a reverse linear pattern, sandwiched between two blocks of single-qubit R_y and R_z rotations, parameterized by $\boldsymbol{\theta} = (\theta_0, ..., \theta_{15})$. Final measurements are performed in the computational basis. The measured bitstrings are used to calculate an average of the cost Hamiltonian $\langle H_C \rangle_{\boldsymbol{\theta}}$. The algorithm aims to minimize $\langle H_C \rangle_{\boldsymbol{\theta}}$ over $\boldsymbol{\theta}$.

is thus left unchanged by V_i, as desired. When starting from a valid state, e.g. the all-zero state, and using this mixer, the system will stay in the space of valid states, so that all quadratic penalty terms can be omitted. The operators V_i associated with different i-s do not necessarily commute, thus their order must be specified. In Paper III, we applied them in a random order.

Hardware efficient ansatz (HEA)

In Paper V, the goal was to try protein design on gate-based hardware using variational quantum algorithms, two of which were QAOA and QAOA+. Both approaches can suffer from having large circuit depths, from either H_C or U_M. High depth poses a challenge for near-term quantum devices due to the accumulation of gate errors. In addition to QAOA and QAOA+, we therefore considered the hardware-effiecent ansatz (HEA) [58], which uses as a shallower circuit tailored to the hardware (Fig 3.5). This algorithm consists of alternating layers of single-qubit rotational gates, parametrized by $\boldsymbol{\theta}$, and CNOTS, arranged in a pattern that suits the hardware.

Unlike QAOA and QAOA+, HEA is problem agnostic; the quantum circuit uses no information about the cost function, H_C. It searches for low-cost solutions by minimizing the average of H_C given $\boldsymbol{\theta}$, $\langle H_C \rangle_{\boldsymbol{\theta}}$, over $\boldsymbol{\theta}$, where $\langle H_C \rangle_{\boldsymbol{\theta}}$ is estimated using measured bitstrings. As in QAOA and QAOA+, the parameter search can be challenging, but suitable parameter-initialization schemes may facilitate the classical optimization.

IBM's gate-based devices

For the gate-based hardware computations in Paper V, we used an IBM Torino device with a 133-qubit Heron processor[6]. The circuits were specified using Qiskit [59], which transpiles them into a form compatible with the native gates and connectivity of the selected device [60]. This includes assigning logical qubits to physical qubits and routing unsupported two-qubit interactions, potentially introducing additional operations such as SWAP gates. The resulting device-compatible circuit is then executed on the device [61].

IBM's gate-based devices use superconducting, transmon-based qubits arranged according to a device-specific coupling map. The IBM Torino used in Paper V, has a heavy-hexagonal connectivity map, where each qubit is connected to at most 3 others [62, 63]. Although readout errors, decoherence characterized by the relaxation and dephasing times T_1 and T_2, and other calibration-dependent errors also contribute [62, 63], in practice two-qubit gate errors are usually among the dominant sources of error. Deep circuits with many two-qubit gates, such as those needed for highly connected H_Cs in QAOA, therefore accumulate errors rapidly.

Quantum error correction

While noise may seem to represent an insurmountable obstacle for quantum computing, a long-term route beyond the noise limitations exists. It is called quantum error correction and is currently the most important research area pushing towards fault-tolerant quantum computers. Quantum error correction works by encoding one logical qubit across many physical qubits. Errors are inferred from repeated measurements without directly measuring the logical state. If the physical error rate is below a threshold, increasing the physical qubits per logical qubit should suppress logical errors exponentially [64–67]. Recent experiments have taken important steps towards this regime [68, 69], while machine-learning decoders have improved identification of errors [70, 71]. However, fault-tolerant devices are, at the time of writing, still out of reach which is why characterizing the performance of present-day noisy devices, as done throughout this thesis, remains important.

[6]As of April 2026, the Torino computer has been retired.

4 Lattice protein problems

The aim of this thesis is to investigate how lattice protein folding, design, and multi-chain thermodynamics, can be formulated and solved on present-day quantum hardware using the computational methods in Ch. 3. In this chapter, these three problems are defined.

Proteins are ubiquitous in living organisms [72]. Their functions are numerous and range from enzymatic catalysis to intracellular cargo transport [72]. The building blocks of proteins are 20 amino acids and each protein is a unique chain of amino acids. In the living cell, many proteins adopt well-defined three-dimensional equilibrium structure [72]. This structure is vital for the function of the protein and depends on the amino acid sequence. Recently, evolutionarily informed machine-learning methods [73, 74] have been developed which can to predict the structure of proteins with great success.

4.1 Lattice proteins

Proteins are immensely complex systems, with a plethora of competing forces giving rise to complicated dynamics and a complex equilibrium distribution. As such, it is often useful to study simplified models of proteins. One widely used approached is to consider lattice-based models, where proteins are modeled as self-avoiding chains of amino acids, or beads, residing on a lattice [75]. Even with this simplification, the number of configurations grows exponentially with the length, N, of the protein [76, 77]. For instance, on a two-dimensional (2D) lattice, the number of configurations scales as μ^N with $\mu \approx 2.6$ [76, 77].

The Hydrophobic-Polar (HP) model

In the Hydrophobic-Polar (HP) model, which we used as test bed in Papers I, III–VI, a lattice protein consists of two types of beads, hydrophobic (H) and polar (P) [75–77]. The interaction potential is pairwise additive and stipulates that two amino acids

interact only if they are nearest neighbors on the lattice but not along the chain, and both are H. Let $\mathbf{r}_i$ and a_i denote, respectively, the position and type (H or P) of amino acid i ($i = 1, \ldots, N$). The interaction energy takes the form [77]

$$E_{\mathrm{HP}} = \sum_{1 \le i < j \le N} C(a_i, a_j)\Delta(\mathbf{r}_i, \mathbf{r}_j), \tag{4.1}$$

where $\Delta(\mathbf{r}_i, \mathbf{r}_j) = 1$ if $\mathbf{r}_i$ and $\mathbf{r}_j$ are nearest neighbors on the lattice, and $\Delta(\mathbf{r}_i, \mathbf{r}_j) = 0$ otherwise. The contact energy is given by $C(a_i, a_j) = -1$ if $a_i = a_j = \mathrm{H}$, and $C(a_i, a_j) = 0$ otherwise. In this convention, the energy of the protein is $-N_{\mathrm{HH}}$, where N_{HH} is the number of nearest-neighbor HH contacts.

In part due to its simplicity, the HP model has been extensively studied [76–79]. In particular, through exhaustive enumeration, all HP sequences that have a unique minimum-energy structure in 2D, and their corresponding structures, have been determined for chains with ≤30 beads [76, 77]. On a 2D square lattice approximately 2% of all HP sequences with ≤30 beads have well-defined minimum-energy structures [76]. These ground-state structures, and other low-energy configurations, tend to display a hydrophobic core of H beads.

The Miyazawa-Jernigan (MJ) model

Another lattice protein model is the Miyazawa-Jernigan (MJ) model. In this model, a 20-letter amino acid alphabet is used (a_i can take one of 20 values). The interaction energy, E_{MJ}, can then be written as

$$E_{\mathrm{MJ}} = \sum_{1 \le i < j \le N} C(a_i, a_j)\Delta(\mathbf{r}_i, \mathbf{r}_j), \tag{4.2}$$

where the contact energies $C(a_i, a_j)$ are given by the statistically derived Miyazawa-Jernigan interaction matrix (Table VI of Ref. [80]). This model has been used to, for instance, understand the dynamics of proteins [81] and to study the basics of the sequence-structure relationship by enumerating configurations on small grids with the same number of lattice points as the size of the chain [82].

4.2 Folding lattice proteins

Protein folding is the process by which a protein goes from an unstable random-coil conformation to its three-dimensional structure [83]. In lattice protein models, all possible structures are represented by discrete conformations on a lattice. Structurally

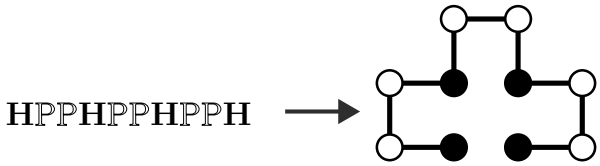


Figure 4.1: Schematic illustration of HP lattice protein folding for the $N = 10$ sequence HPPHPPHPPH. Given a sequence of Hs (●) and Ps (○) beads (left), the task is to minimize E_{HP} (or E_{MJ}) in Eq. (4.1) [or (4.2)]. The solution is the structure on the right.

characterizing a lattice protein requires the use of an ensemble, where the probability of the structure C is given by

$$P(C \mid \boldsymbol{s}) = e^{-\beta E(C,\boldsymbol{s})} \Big/ \sum_{C'} e^{-\beta E(C',\boldsymbol{s})}. \tag{4.3}$$

Here $\boldsymbol{s} = (a_1, .., a_N)$ is the amino acid sequence and $E(C, \boldsymbol{s})$ is the energy of conformation C, either E_{HP} or E_{MJ}, for $\boldsymbol{s}$.

However, many real proteins adopt a largely well-defined native structure. An entry-level proxy to describing the full folding process is identifying this native state, given the amino acid sequence. For lattice protein models, this amounts to searching for the minimum-energy structures C^* (Fig. 4.1), defined as

$$C^* = \arg\min_{C} E(C, \boldsymbol{s}). \tag{4.4}$$

This optimization problem is called structure prediction, but is, for brevity, also referred to as folding. Papers I–III deal with this optimization problem.

4.3 Designing lattice proteins

In lattice protein design, one seeks an amino acid sequence $\boldsymbol{s} = (a_1, \ldots, a_N)$ that folds into a given target structure C_t. See Fig. 4.2 for an illustration. To this end, the goal is to maximize the probability of finding the chain in the state C_t, i.e. to find

$$\boldsymbol{s}^* = \arg\max_{\boldsymbol{s}} P(C_t \mid \boldsymbol{s}), \tag{4.5}$$

where P is defined in Eq. (4.3). Solving Eq. (4.5) generally involves a time-consuming search in both sequence and structure spaces. Although sophisticated methods for this task have been developed [84–86], a common approach is to ignore the sum in

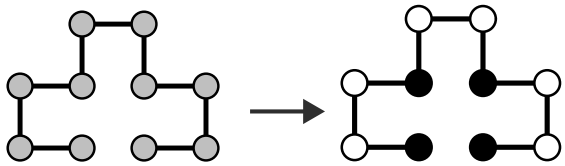


Figure 4.2: Illustration of designing an HP lattice protein of length $N = 10$. Given a structure (left), the task is to find the sequence that folds into this structure. The solution is a sequence of H (●) and P (○) beads (right).

Eq. (4.3) and minimize the energy inside the target structure, $E(C_t, \boldsymbol{s})$ [37, 51, 87, 88] or to introduce an approximation $e^{-\beta\tilde{F}(\boldsymbol{s})}$ of the sum and minimize $E(C_t, \boldsymbol{s}) - \tilde{F}(\boldsymbol{s})$ [89]. This approaches turns the design problem into a simpler optimization problem. However, an additional filtering step is needed to reject candidate sequences that have a higher probability for some other structure $C \neq C_t$, which is often done by folding the candidate sequence [37].

4.4 Multi-chain systems

While many real proteins have a unique equilibrium structure that determines their biological function, many proteins also lack a well-defined three-dimensional structure. For instance, in the human genome, it has been estimated that 44 % of the encoded proteins contain stretches of disorder longer than 30 amino acids [90]. Disordered proteins often participate in multi-chain processes, such as droplet condensation and aggregation, which have been linked to several neurodegenerative diseases [91, 92]. For such systems, finding a single ground-state structure is less desirable and the problem shifts to characterizing the thermodynamics of the system.

Disordered proteins are believed to often play a driving role in the formation of membraneless organelles in the cell [91]. The propensity of disordered proteins to form a droplets is known to depend on their sequences [91]. To better understand such systems, estimating phase diagrams and their sequence-dependence is an important but difficult task. Thus, lattice-based and other coarse-grained models have a role to play. To estimate these phase diagrams for lattice proteins, one needs to sample multi-chain configurations, C, using Eq. (4.3), with $\boldsymbol{s}$ replaced by the sequences of every chain in the system.

5 Binary representations of lattice protein problems

To use the quantum algorithms introduced in Ch. 3, the problem of interest needs to be recast to an Ising or QUBO form. In this section, I introduce QUBO encodings of the lattice protein folding, design and multi-chain thermodynamics problems of Ch. 4. The QUBO encodings below are written using bits, $\boldsymbol{b} \in \{0,1\}^n$. A QUBO energy function $E(\boldsymbol{b})$ can be easily transformed to a quantum Hamiltonian, $H_C(\boldsymbol{\sigma}^z)$, by the transformation $b_i \to (1-\sigma_i^z)/2$.

5.1 Lattice protein folding

To recast lattice protein folding into a QUBO form, one first needs to decide what the binary variables represent. Given this, the second aim becomes to construct an energy function whose ground state corresponds to the ground-state structure of the lattice protein sequence. This involves incorporating soft constraints in the energy function, such that the generated solutions correspond to physical chains.

Turn-based encoding

In the turn-based encodings, turns along the chain are mapped onto the binary variables [93–95]. These encodings are resource-efficient for small chains since they offer a compact chain representation by only using $O(N)$ variables for an N-long chain. However, the soft constraints of the model, such as self-avoidance, are difficult to implement and many auxiliary variables are required to make the resulting energy function quadratic. The additional variables negate the resource efficiency of the encoding provided by the chain representation, resulting in $O(N^2)$ binary variables [96].

The earliest attempt to fold a short lattice protein ($N = 6$) using quantum annealing carried out on a D-Wave machine used a turn-based encoding [93]. Similar encodings have been implemented on IBM's gate-based quantum computers [97, 98].

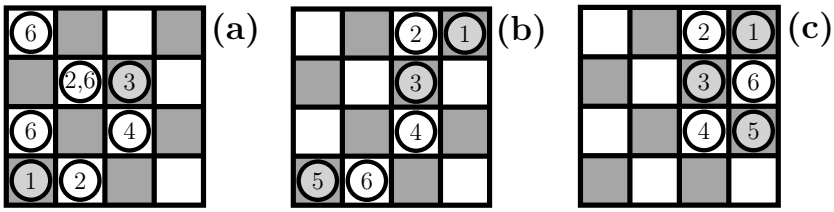


Figure 5.1: Illustration of a hypothetical SA minimization of Eqs. (5.2)–(5.5) for the 6-bead sequence HPHPPH. Circles represent beads, and numbers indicate bead positions along the sequence. By construction, odd/even beads can reside only on grey/white sites. (a) Early stage. Typically, all three constraints are violated ($E_1, E_2, E_3 > 0$). (b) Intermediate stage. Some but not all of the constraints are satisfied (in this example: $E_1 = E_2 = 0$, $E_3 > 0$). (c) The final state, in this example corresponding to the desired minimum-energy structure of the given sequence ($E_{\mathrm{HP}} = -2$, $E_1 = E_2 = E_3 = 0$).

Coordinate-based encodings

In coordinate-based encodings, the binary variables represent the position of the beads on the lattice. An early attempt was made to construct an energy function where the binary variables define integer coordinates of each amino acid. Unfortunately, the resulting energy function required $O(N^3)$ bits, after introducing the necessary auxiliary variables to make the function quadratic [99].

In Paper I, we instead developed a field-like coordinate-based encoding of 2D lattice protein folding, inspired by encodings of homopolymers [100, 101], that is entirely quadratic and designed to handle large proteins. Paper II expanded the encoding to 3D. The representation uses bits $b_{i,s}$ that indicate whether amino acid i is located on lattice site s ($b_{i,s} = 1$) or not ($b_{i,s} = 0$). The energy function is given by

$$E = E_{\mathrm{LP}} + \lambda_1 E_1 + \lambda_2 E_2 \ + \lambda_3 E_3, \tag{5.1}$$

where E_{LP} is the physical energy of the lattice protein model (e.g. E_{HP} or E_{MJ} from Ch. 4), and the remaining three terms E_1, E_2 and E_3 are soft constraints that ensure that the bit configurations correspond to valid chain structures. The strengths of the penalty terms are set by the Lagrange parameters λ_i. All four energy terms in Eq. (5.1) have quadratic closed-form expressions valid for any chain length, which are as follows [102, 103].

- The physical energy E_{LP} can be written as

$$E_{\mathrm{LP}} = \sum_{|i-j|>1} C(a_i, a_j) \sum_{\langle s,s' \rangle} b_{i,s} b_{j,s'}, \tag{5.2}$$

where $\langle \cdot, \cdot \rangle$ indicates summation over nearest-neighbor pairs of sites and $C(a_i, a_j)$ are the contact energies of the chosen model [e.g Eq. (4.1) or (4.2)].

- The first penalty energy, E_1, is given by

$$E_1 = \sum_i \left(\sum_s b_{i,s} - 1 \right)^2 = \sum_i \left(\sum_{s \neq s'} b_{i,s} b_{i,s'} - \sum_s b_{i,s} + 1 \right), \tag{5.3}$$

  and serves to ensure that each bead is located at exactly one lattice site.

- The energy E_2 enforces chain self-avoidance, and is given by

$$E_2 = \sum_s \sum_{i<j} b_{i,s} b_{j,s}. \tag{5.4}$$

- The third and final penalty energy, E_3, is responsible for connecting the beads to a chain, and can be written as

$$E_3 = \sum_{1 \leq i < N} \sum_s b_{i,s} \sum_{\|s'-s\|>1} b_{i+1,s'}, \tag{5.5}$$

  where $\|s' - s\|$ denotes the Euclidean distance between the sites s' and s (assuming unit lattice spacing).

Note that the Eqs. (5.3)–(5.5) are densely connected. For a chain with N amino acids on an L^2 grid, this mapping requires $n = NL^2$ bits. With a checkerboard division of the grid into odd and even sites, it is often possible to reduce the bit count to $n \approx NL^2/2$ [see Fig. 5.1 for a hypothetical simulated annealing minimization of Eq. (5.1)]. Linn et al. performed resource analysis of both the field-like encoding presented here and turn-based encodings. They found that while turn-based encodings use fewer qubits for short chains, the field-like one was more resource efficient for larger chains [96].

5.2 Lattice protein design

In the design problem, both sequence and structure space needs to be explored. QUBO formulations are thus broken down into a two-step procedure [37, 89]. Given a target structure C_t, the two steps are

1. Find a sequence, $\boldsymbol{s}$, that minimizes the energy $E(C_t, \boldsymbol{s})$ in the target structure.
2. Fold the obtained sequence to make sure it does not fold to a different structure $C \neq C_t$.

Figure 5.2: Illustration of the process to design an HP lattice protein of length $N = 10$ adopting the structure in figs. 4.1 and 4.2, using the binary representation in sec 5.2. Step 1 consists of finding the optimal sequence by minimizing Eq. (5.6) using the connectivity matrix w_{ij}. The second step is folding the obtained sequence, to make sure it folds into the target structure. Here H beads are represented as ● and P beads are represented as ○.

For step 2, one can use the binary QUBO mapping above. Most treatments of these two steps address only step 1 using quantum optimization [51, 87–89]. In Paper IV we instead carried out both steps using HA.

To perform step 1, one introduces binary variables b_i for each amino acid. In the HP model this is particularly simple: $b_i = 0$ represents P and $b_i = 1$ represents H. The energy function then becomes

$$E(\boldsymbol{s}) = -\sum_{1\leq i<j\leq N} w_{ij} b_i b_j + \text{soft constraints}, \tag{5.6}$$

where w_{ij} is the connectivity matrix in the structure. It is $w_{ij} = 1$ when the sites i and j are nearest neighbors on the lattice but not along the chain (see Fig. 5.2).

The soft constraints needed in Eq. (5.6) depend on the model used. For the HP model, this constraint is often set to

$$\lambda\left(\sum_i b_i - N_{\mathrm{H}}\right)^2, \tag{5.7}$$

which biases the total number of H-type beads toward a preset value, N_{H}. λ sets the strength of this constraint. Similarly to Eq. (5.3), this soft constraint is highly connected, increasing the hardware overhead required to enforce physical solutions.

For models with K amino acid types, one often resorts to a one-hot or domain-wall encoding [104]. In both cases, the qubit count per site increases from one to $O(K)$, leading to a total of $O(KN)$ qubits. In these cases, $-b_i b_j$ is replaced by the models contact energy, $C(a_i, a_j)$.

5.3 Multi-chain systems

The QUBO encoding of multi-chain systems builds on the folding encoding of Sec. 5.1. The same field-like bits are used but the chain-connectivity term E_3 is restricted to act only within chains. The major change for multi-chain systems is the goal. Instead of finding a (single) ground-state structure, obtaining thermodynamic quantities is the ultimate goal.

Slongo and Micheletti developed a weighted histogram analysis method for QUBO problems [10] (see Sec. 2.1). In Paper VI, we extended this method to multi-chain HP systems using additional machinery in the form of auxiliary bits a_k, link variables $\eta_{ss'}$, and a new penalty term E_4 to generate estimates for the density of states $g(E_{\mathrm{HP}})$. Specifically, the machinery requires the physical energy term [Eq. (5.2)] to be constrained to a certain interval. To sample states in the energy interval $E^- \leq E_{\mathrm{HP}} \leq E^+ = E^- + r - 1$, the physical energy E_{HP} is replaced in the QUBO objective function by

$$E_{int} = \left(E_{\mathrm{HP}} - E^- - \sum_{k=0}^{r-1} k a_k \right)^2 + \left(\sum_{k=0}^{r-1} a_k - 1 \right)^2 , \tag{5.8}$$

in the existing energy function. Here $a_0, \ldots, a_{r-1}$ are r auxiliary bits that make sure the physical energy is within the desired interval, which add r additional bits to the total energy function.

Since Eq. (5.2) is quadratic, constraining it to be in a certain interval introduces cubic and quartic terms. To keep the energy function quadratic, Eq. (5.2) is linearized by introducing auxiliary link variables $\eta_{ss'}$, associated with nearest-neighbor pairs of lattice points s and s', giving the linear form

$$E'_{\mathrm{HP}} = - \sum_{\langle s,s' \rangle} \eta_{ss'} . \tag{5.9}$$

This can straightforwardly replace E_{HP} in Eq. (5.8), and the final expression is still at most quadratic. For an $L \times L$ grid, this construction requires $2L(L-1)$ additional bits. A link $\eta_{ss'}$ is defined as active ($\eta_{ss'} = 1$) if both s and s' are occupied by an H bead, and inactive ($\eta_{ss'} = 0$) otherwise (see Fig. 5.3 for an illustration).

To make sure $\eta_{ss'} = 1$ when both s and s' are occupied by H beads and 0 otherwise,

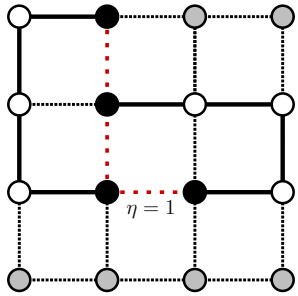


Figure 5.3: Illustration of the minimum-E_{HP} configuration for a system of two HPPPH chains. H beads are represented as ● and P beads are represented as ○, whereas the grey circles indicate empty sites. The three active links ($\eta_{ss'} = 1$) are indicated by red, loosely dashed lines. The dotted lines show inactive links ($\eta_{ss'} = 0$) and solid black lines show the protein chain.

an additional penalty energy is needed, E_4. This penalty is defined as

$$E_4 = \sum_{\langle s,s' \rangle} \Bigg[3\eta_{ss'} + \sum_{\substack{(\alpha,f)\neq(\alpha',f') \\ f,f' \in \mathcal{H}}} b_s^{(\alpha,f)} b_{s'}^{(\alpha',f')} - 2\eta_{ss'} \sum_{\alpha} \sum_{f \in \mathcal{H}} \left(b_s^{(\alpha,f)} + b_{s'}^{(\alpha,f)} \right) \Bigg], \tag{5.10}$$

where α is a chain index and f indicates the position along a chain. Assuming that the other constraints corresponding to the penalty energies E_1, E_2 and E_3 are met, it can be easily verified that the expression within square brackets in Eq. (5.10) is nonnegative. It takes its minimum value zero for $\eta_{ss'} = 1$ if both s and s' are occupied by H beads, and for $\eta_{ss'} = 0$ otherwise.

With E_4 included, our final QUBO objective function E_c for interval sampling consists of five terms and depends on three sets of bits, namely $\mathbf{b} = (\ldots, b_s^{(\alpha,f)}, \ldots)$, $\boldsymbol{\eta} = (\ldots, \eta_{ss'}, \ldots)$ and $\mathbf{a} = (a_0, \ldots, a_{r-1})$. Specifically, it has the form

$$E_c(\mathbf{b}, \boldsymbol{\eta}, \mathbf{a}) = \lambda_0 E_{int}(\boldsymbol{\eta}, \mathbf{a}) + \lambda_1 E_1(\mathbf{b}) + \lambda_2 E_2(\mathbf{b}) + \lambda_3 E_3(\mathbf{b}) + \lambda_4 E_4(\mathbf{b}, \boldsymbol{\eta}), \tag{5.11}$$

and requires $MNL^2 + 2L(L-1) + r$ total bits to implement. With E_c defined, generating an estimate of $g(E_{\mathrm{HP}})$ for a given system amounts to repeatedly minimizing Eq. (5.11) over different but overlapping intervals $[E^-, E^+]$ and combining the results via a weighted multi-histogram analysis (Sec. 2.1). Thermodynamic quantities can subsequently be computed at arbitrary temperatures.

6 Summary of the papers & outlook

As quantum computers move from a theoretical idea to real devices, an important task is understanding how they perform on concrete and challenging problems. Since analytical performance guarantees remain out of grasp for all but a few landmark cases, the most direct way to gain this understanding is to implement the problem and evaluate its performance in practice. Lattice protein problems (Ch. 4), which have a simple mathematical structure while retaining computational difficulties encountered for real proteins, represent such concrete and challenging problems.

To illustrate how these problems can be implemented and tested on hardware, the first part of this thesis introduced binary representations of the problems (Ch. 5), as well as quantum (Ch. 3) and classical (Ch. 2) optimization methods. In this chapter, I provide a summary of the papers included in the thesis, Papers I–VI, and an outlook.

Prior to my first contributions, quantum hardware demonstrations of lattice protein folding were limited to short chains ($N \leq 6$ for annealers, $N \leq 7$ for gate-based hardware) using turn-based encodings [93, 95]. Moreover, implementations of the design problem only did step 1 [87, 88] and multi-chain thermodynamics had not been demonstrated on quantum hardware at all. The six papers summarized below addressed these gap and, in doing so, characterize where the algorithms in Ch. 3 succeed and where they struggle.

Paper I. We created the two-local mapping from sec. 5.1 and implemented it on a D-Wave quantum annealer. Prior quantum annealing demonstrations had folded chains of at most $N = 6$ [93], and the mappings used grew in complexity when increasing N. Our mapping, designed for large chains, combined with HA, folded 2D HP lattice proteins of up to 64 beads. On the pure QPU, however, success rates decreased exponentially, limiting folding to chains with $N \leq 14$. We attributed the poor QPU performance to integrated control errors amplified by long embedding chains, required by the high connectivity of H_C and largely caused by the soft constraints in Eqs. (5.3)–(5.5).

Paper II. We extended the mapping in Sec. 5.1 to 3D MJ lattice proteins on minimal

grids, which have one lattice point per bead. This minimal-grid regime had not previously been studied using quantum annealing. Such systems are difficult for conventional methods due to steric clashes. Our mapping combined with HA swiftly folded them, whereas the commercial optimizer Gurobi, based on linear and quadratic programming [105], could not solve the problem. Our results showed that our approach extends beyond the 2D HP setting of Paper I and that the soft constraints of the model can sometimes be beneficial because temporary constraint violations may permit unconventional paths through configuration space.

Paper III. The successes and limitations of quantum annealing motivated us to extend the framework to gate-based architectures, for which the experimental progress is faster. However, direct implementations of lattice protein QUBOs rapidly become intractable because of growing qubit and gate counts [96], largely caused by the soft constraint terms. By mapping the binary variables to nodes and the quadratic terms in the soft constraints [Eqs. (5.3)–(5.5)] to edges in a graph, we used the mixer circuit in Eq. (3.22) to remove the quadratic constraint terms entirely. Classical circuit simulations showed our new mixer approach performs favorably compared to standard QAOA on proteins of length $N = 4$ and 6, and a divide-and-conquer heuristic built on the same mixer let us fold proteins up to $N = 14$ using at most 26 qubits.

Paper IV. Having studied folding, we next considered the inverse problem, lattice protein design. We presented the two-step procedure described in Sec. 5.2. Using HA, we found candidate sequences and verified that they folded uniquely to the target structures. The lower qubit requirements of step 1 also allowed us to investigate the exponential success-rate decay observed in Paper I. A simple noise model [Eq. (3.9)] reproduced the results semiquantitatively. The soft constraint increased problem connectivity and thus required longer, more strongly coupled embedded chains and thus greater rescaling, amplifying integrated control errors relative to the target Hamiltonian. This suggests that the same mechanism limits both folding and design.

Paper V. We revisited step 1 of the design problem using the gate-based algorithms QAOA, QAOA+ and HEA, providing the first comparison of these algorithms on lattice protein design under realistic noise. Classical circuit simulations showed that QAOA+ with the XY-mixer performs well without noise but degrades once noise is included, due to its large circuit depth. In comparison, HEA performed better under noise despite discarding problem-specific structure. Running HEA on an IBM device, using parameters from simulation, yielded a success probability of at least 0.2 in most instances, which was higher than what we achieved using the pure QPU in Paper IV. This constituted, to our knowledge, the first lattice protein design result obtained using variational quantum algorithms on physical gate-based hardware.

Paper VI. We extended our mapping in Paper I from ground-state search for single

chains to the thermodynamics of lattice protein systems, which to our knowledge had not previously been studied using quantum optimization. The closest related work is a weighted multi-histogram analysis scheme for QUBO-encoded lattice ring-polymer melts [10]. We extended this approach using the mapping in Sec. 5.3 and reconstructed densities of states and equilibrium properties with HA. The resulting densities showed system-dependent over- and undersampling of low-energy states. We attributed this to biased sampling within the degenerate ground state of Eq. (5.11). Analysis of a single-chain system showed that the bias depended strongly on the Lagrange parameters. Despite this, the reconstructed thermodynamics retained meaningful features.

Taken together, the six papers show that three main lattice protein problems, namely folding, design, and multi-chain thermodynamics, can be formulated as binary optimization problems. All three build on the mapping introduced in Paper I, extending it from ground-state folding to design and thermodynamics. Although classical methods still reach larger systems, current hybrid platforms can already handle substantial instances of these challenging problems, while pure-QPU results are more modest. Our results represent progress toward classically relevant problem sizes.

Papers I, IV, V, and VI show that a recurrent problem lies in the soft constraints used to enforce physical solutions, which increase problem connectivity. In quantum annealing, this leads to longer embedding chains, which in turn require stronger interactions between physical qubits and thus a stronger rescaling. This amplifies the strength of integrated control errors relative to the user-given Hamiltonian. On gate-based hardware, they increase the circuit depth of QAOA and therefore the sensitivity to noise. When sampling using HA, the observed bias showed a strong dependence on the Lagrange parameters. However, soft constraints are not always detrimental. In Paper II, temporary constraint violations may have allowed the solver to avoid steric barriers.

Several approaches could reduce constraint costs while retaining the freedom of temporary violations. Papers III and V partly explored this through constraint-preserving mixers and shallower hardware-efficient circuits. The challenge is to balance efficient optimization, low circuit depth, and sufficient freedom to explore unphysical intermediate states. Further progress will likely require combining improved algorithms and problem formulations.

Hardware development is equally important. In quantum annealing, limited QPU connectivity requires long embedding chains, which amplify analog control errors. Improved connectivity and control precision would directly address a main bottleneck identified in this thesis. For gate-based hardware, reducing two-qubit gate errors would permit deeper circuits and make problem-informed algorithms practical. Together, these developments could bring quantum methods for lattice proteins closer to classically competitive scales and enable their extension to more realistic models.

Bibliography

[1] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information: 10th Anniversary Edition* (Cambridge University Press, 2010).

[2] P. Shor, Algorithms for quantum computation: discrete logarithms and factoring, in *Proceedings 35th Annual Symposium on Foundations of Computer Science* (1994) pp. 124–134.

[3] P. W. Shor, Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, SIAM J. Comput. **26**, 1484 (1997).

[4] L. K. Grover, A Fast Quantum Mechanical Algorithm for Database Search, in *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '96 (Association for Computing Machinery, New York, NY, USA, 1996) pp. 212–219.

[5] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, Optimization by simulated annealing, Science **220**, 671 (1983).

[6] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, Equation of state calculations by fast computing machines, J. Chem. Phys. **21**, 1087 (1953).

[7] W. K. Hastings, Monte Carlo sampling methods using Markov chains and their applications, Biometrika **57**, 97 (1970).

[8] A. M. Ferrenberg and R. H. Swendsen, New monte carlo technique for studying phase transitions, Phys. Rev. Lett. **61**, 2635 (1988).

[9] A. M. Ferrenberg and R. H. Swendsen, Optimized monte carlo data analysis, Phys. Rev. Lett. **63**, 1195 (1989).

[10] F. Slongo and C. Micheletti, Computing canonical averages with quantum and classical optimizers: Thermodynamic reweighting for qubo models of physical systems, Phys. Rev. Res. **7**, 023116 (2025).

[11] F. Wang and D. P. Landau, Efficient, multiple-range random walk algorithm to calculate the density of states, Phys. Rev. Lett. **86**, 2050 (2001).

[12] A. Lucas, Ising formulations of many NP problems, Front. Phys. **2**, 1 (2014).

[13] S. Geman and D. Geman, Stochastic relaxation, gibbs distributions, and the bayesian restoration of images, IEEE Trans. Pattern Anal. Mach. Intell. **PAMI-6**, 721 (1984).

[14] S. Tanaka, R. Tamura, and B. K. Chakrabarti, *Quantum spin glasses, annealing and computation* (Cambridge University Press, 2017).

[15] M. Willsch, D. Willsch, and K. Michielsen, Lecture notes: Programming quantum computers, arXiv:2201.02051 (2022).

[16] A. F. Kockum, A. Soro, L. García-Álvarez, P. Vikstål, T. Douce, G. Johansson, and G. Ferrini, Lecture notes on quantum computing, arXiv:2311.08445 (2025).

[17] T. Albash and D. A. Lidar, Adiabatic quantum computation, Rev. Mod. Phys. **90**, 015002 (2018).

[18] E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda., A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem, Science **292**, 472 (2001).

[19] T. Kadowaki and H. Nishimori, Quantum annealing in the transverse Ising model, Phys. Rev. E **58**, 5355 (1998).

[20] M. Born and V. Fock, Beweis des adiabatensatzes, Z. Phys. **51**, 165 (1928).

[21] A. Rajak, S. Suzuki, A. Dutta, and B. K. Chakrabarti, Quantum annealing: an overview, Philos. Trans. R. Soc. A **381**, 20210417 (2022).

[22] V. Bapst, L. Foini, F. Krzakala, G. Semerjian, and F. Zamponi, The quantum adiabatic algorithm applied to random optimization problems: The quantum spin glass perspective, Phys. Rep. **523**, 127 (2013), the Quantum Adiabatic Algorithm Applied to Random Optimization Problems: The Quantum Spin Glass Perspective.

[23] M. Bernaschi, I. González-Adalid Pemartín, V. Martín-Mayor, and G. Parisi, The quantum transition of the two-dimensional ising spin glass, Nature **631**, 749 (2024).

[24] A. D. King, J. Raymond, T. Lanting, R. Harris, A. Zucca, F. Altomare, A. J. Berkley, K. Boothby, S. Ejtemaee, C. Enderud, E. Hoskinson, S. Huang, E. Ladizinsky, A. J. R. MacDonald, G. Marsden, R. Molavi, T. Oh, G. Poulin-Lamarre, M. Reis, C. Rich, Y. Sato, N. Tsai, M. Volkmann, J. D. Whittaker, J. Yao, A. W. Sandvik, and M. H. Amin, Quantum critical dynamics in a 5,000-qubit programmable spin glass, Nature **617**, 61 (2023).

[25] A. D. King, A. Nocera, M. M. Rams, J. Dziarmaga, R. Wiersema, W. Bernoudy, J. Raymond, N. Kaushal, N. Heinsdorf, R. Harris, K. Boothby, F. Altomare, M. Asad, A. J. Berkley, M. Boschnak, K. Chern, H. Christiani, S. Cibere, J. Connor, M. H. Dehn, R. Deshpande, S. Ejtemaee, P. Farre, K. Hamer, E. Hoskinson, S. Huang, M. W. Johnson, S. Kortas, E. Ladizinsky, T. Lanting, T. Lai, R. Li, A. J. R. MacDonald, G. Marsden, C. C. McGeoch, R. Molavi, T. Oh, R. Neufeld, M. Norouzpour, J. Pasvolsky, P. Poitras, G. Poulin-Lamarre, T. Prescott, M. Reis, C. Rich, M. Samani, B. Sheldan, A. Smirnov, E. Sterpka, B. T. Clavera, N. Tsai, M. Volkmann, A. M. Whiticar, J. D. Whittaker, W. Wilkinson, J. Yao, T. J. Yi, A. W. Sandvik, G. Alvarez, R. G. Melko, J. Carrasquilla, M. Franz, and M. H. Amin, Beyond-classical computation in quantum simulation, Science **388**, 199 (2025).

[26] H. Munoz-Bauza and D. Lidar, Scaling advantage in approximate optimization with quantum annealing, Phys. Rev. Lett. **134**, 160601 (2025).

[27] C. L. Baldwin, Reaching states below the threshold energy in spin glasses via quantum annealing, arXiv:2603.23602 (2026).

[28] R. Harris, T. Lanting, A. J. Berkley, J. Johansson, M. W. Johnson, P. Bunyk, E. Ladizinsky, N. Ladizinsky, T. Oh, and S. Han, Compound josephson-junction coupler for flux qubits with minimal crosstalk, Phys. Rev. B **80**, 052506 (2009).

[29] R. Harris, J. Johansson, A. J. Berkley, M. W. Johnson, T. Lanting, S. Han, P. Bunyk, E. Ladizinsky, T. Oh, I. Perminov, E. Tolkacheva, S. Uchaikin, E. M. Chapple, C. Enderud, C. Rich, M. Thom, J. Wang, B. Wilson, and G. Rose, Experimental demonstration of a robust and scalable flux qubit, Phys. Rev. B **81**, 134510 (2010).

[30] R. Harris, M. W. Johnson, T. Lanting, A. J. Berkley, J. Johansson, P. Bunyk, E. Tolkacheva, E. Ladizinsky, N. Ladizinsky, T. Oh, F. Cioata, I. Perminov, P. Spear, C. Enderud, C. Rich, S. Uchaikin, M. C. Thom, E. M. Chapple, J. Wang, B. Wilson, M. H. S. Amin, N. Dickson, K. Karimi, B. Macready, C. J. S. Truncik, and G. Rose, Experimental investigation of an eight-qubit unit cell in a superconducting optimization processor, Phys. Rev. B **82**, 024511 (2010).

[31] D-Wave, D-wave ocean software documentation (2026), accessed: 2026-07-23.

[32] D-Wave, Error sources (2026), accessed: 2026-07-23.

[33] A. Pearson, A. Mishra, I. Hen, and D. A. Lidar, Analog errors in quantum annealing: doom and hope, Npj Quantum Inf. **5**, 107 (2019).

[34] V. Martin-Mayor and I. Hen, Unraveling quantum annealers using classical hardness, Sci. Rep. **5**, 15324 (2015).

[35] T. Albash, V. Martin-Mayor, and I. Hen, Analog errors in ising machines, Quantum Sci. Technol. **4**, 02LT03 (2019).

[36] D-Wave, Properties of the quantum processing units (2026), accessed: 2026-07-23.

[37] A. Irbäck, L. Knuthson, S. Mohanty, and C. Peterson, Using quantum annealing to design lattice proteins, Phys. Rev. Res. **6**, 013162 (2024).

[38] B. Camino, J. Buckeridge, N. Chancellor, C. R. A. Catlow, A. M. Ferrari, P. A. Warburton, A. A. Sokol, and S. M. Woodley, Exploring the thermodynamics of disordered materials with quantum computing, Sci. Adv. **11**, 10.1126/sciadv.adt7156 (2025).

[39] G. Teza, F. Campaioli, M. Avesani, and O. Raz, Finite-temperature criticality through quantum annealing, Nat. Commun. 10.1038/s41467-026-75348-3 (2026).

[40] P. Sathe, A. D. King, S. M. Mniszewski, C. Coffrin, C. Nisoli, and F. Caravelli, Classical criticality via quantum annealing, Nat. Commun. **17**, 856 (2026).

[41] M. H. Amin, E. Andriyash, J. Rolfe, B. Kulchytskyy, and R. Melko, Quantum boltzmann machine, Phys. Rev. X **8**, 021050 (2018).

[42] J. Nelson, M. Vuffray, A. Y. Lokhov, T. Albash, and C. Coffrin, High-quality thermal Gibbs sampling with quantum annealing hardware, Phys. Rev. Appl. **17**, 044046 (2022).

[43] C. McGeoch and P. Farré, *The D-Wave Advantage System: An Overview*, D-Wave Inc. (2020).

[44] D-Wave, Minor embeddings (2026), accessed: 2026-07-23.

[45] P. Date, R. Patton, C. Schuman, and T. Potok, Efficiently embedding QUBO problems on adiabatic quantum computers, Quantum Inf. Process. **18**, 117 (2019).

[46] C. McGeoch, P. Farré, and W. Bernoudy, *D-Wave Hybrid Solver Service + Advantage: technology update*, Tech. Rep. (D-Wave Systems Inc., 2020).

[47] E. Farhi, J. Goldstone, and S. Gutmann, A quantum approximate optimization algorithm, arXiv:1411.4028 (2014).

[48] L. Zhou, S.-T. Wang, S. Choi, H. Pichler, and M. D. Lukin, Quantum approximate optimization algorithm: Performance, mechanism, and implementation on near-term devices, Phys. Rev. X **10**, 021067 (2020).

[49] I. Lyngfelt and L. García-Álvarez, Symmetry-informed transferability of optimal parameters in the quantum approximate optimization algorithm, Phys. Rev. A **111**, 022418 (2025).

[50] S. Hadfield, Z. Wang, B. O'Gorman, E. G. Rieffel, D. Venturelli, and R. Biswas, From the quantum approximate optimization algorithm to a quantum alternating operator ansatz, Algorithms **12**, 34 (2019).

[51] H. Linn, L. Knuthson, A. Irbäck, S. Mohanty, L. García-Álvarez, and G. Johansson, Designing lattice proteins with variational quantum algorithms, Phys. Rev. Appl. (2026).

[52] L. Gellersen, A. Irbäck, L. Knuthson, and S. Prestel, Penalty-free quantum optimization applied to lattice protein folding, arXiv:2606.02104 (2026).

[53] I. Hen and F. M. Spedalieri, Quantum annealing for constrained optimization, Phys. Rev. Appl. **5**, 034007 (2016).

[54] K. Kawasaki, Diffusion constants near the critical point for time-dependent ising models. i, Phys. Rev. **145**, 224 (1966).

[55] A. Bärtschi and S. Eidenbenz, Deterministic preparation of dicke states, in *Fundamentals of Computation Theory*, edited by L. A. Gąsieniec, J. Jansson, and C. Levcopoulos (Springer, Cham, 2019) pp. 126–139.

[56] T. Tomesh, Z. H. Saleem, and M. Suchara, Quantum Local Search with the Quantum Alternating Operator Ansatz, Quantum **6**, 781 (2022).

[57] Z. H. Saleem, T. Tomesh, B. Tariq, and M. Suchara, Approaches to constrained quantum approximate optimization, SN Comput. Sci. **4**, 183 (2023).

[58] A. Kandala, A. Mezzacapo, K. Temme, M. Takita, M. Brink, J. M. Chow, and J. M. Gambetta, Hardware-efficient variational quantum eigensolver for small molecules and quantum magnets, Nature **549**, 242 (2017).

[59] A. Javadi-Abhari, M. Treinish, K. Krsulich, C. J. Wood, J. Lishman, J. Gacon, S. Martiel, P. D. Nation, L. S. Bishop, A. W. Cross, B. R. Johnson, and J. M. Gambetta, Quantum computing with Qiskit, arXiv:2405.08810 (2024).

[60] IBM Quantum, Introduction to transpilation (2026), accessed: 2026-06-20.

[61] IBM Quantum, Development workflow (2026), accessed: 2026-06-20.

[62] IBM Quantum, View backend details (2026), accessed: 2026-06-20.

[63] IBM Quantum, Torino documention (2026), accessed: 2026-06-20.

[64] P. W. Shor, Scheme for reducing decoherence in quantum computer memory, Phys. Rev. A **52**, R2493 (1995).

[65] A. M. Steane, Error correcting codes in quantum theory, Phys. Rev. Lett. 77, 793 (1996).

[66] A. Y. Kitaev, Quantum computations: algorithms and error correction, Russ. Math. Surv. **52**, 1191 (1997).

[67] S. B. Bravyi and A. Y. Kitaev, Quantum codes on a lattice with boundary, arXiv:quant-ph/9811052 (1998).

[68] Google Quantum AI, Suppressing quantum errors by scaling a surface code logical qubit, Nature **614**, 676 (2023).

[69] Google Quantum AI and Collaborators, Quantum error correction below the surface code threshold, Nature **638**, 920 (2025).

[70] J. Bausch, A. W. Senior, F. J. H. Heras, T. Edlich, A. Davies, M. Newman, C. Jones, K. Satzinger, M. Y. Niu, S. Blackwell, G. Holland, D. Kafri, J. Atalaya, C. Gidney, D. Hassabis, S. Boixo, H. Neven, and P. Kohli, Learning high-accuracy error decoding for quantum processors, Nature **635**, 834 (2024).

[71] M. Lange, P. Havström, B. Srivastava, I. Bengtsson, V. Bergentall, K. Hammar, O. Heuts, E. van Nieuwenburg, and M. Granath, Data-driven decoding of quantum error correcting codes using graph neural networks, Phys. Rev. Res. 7, 023181 (2025), 2307.01241 .

[72] R. Phillips, J. Kondev, J. Theriot, and H. G. Garcia, *Physical Biology of the Cell*, 2nd ed. (Garland Science, New York and London, 2013).

[73] M. Varadi, S. Anyango, M. Deshpande, S. Nair, C. Natassia, G. Yordanova, D. Yuan, O. Stroe, G. Wood, A. Laydon, A. Žídek, T. Green, K. Tunyasuvunakool, S. Petersen, J. Jumper, E. Clancy, R. Green, A. Vora, M. Lutfi, M. Figurnov, A. Cowie, N. Hobbs, P. Kohli, G. Kleywegt, E. Birney, D. Hassabis, and S. Velankar, AlphaFold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models, Nucleic Acids Res. **50**, D439 (2021).

[74] J. Jumper, R. Evans, A. Pritzel, T. Green, M. Figurnov, O. Ronneberger, K. Tunyasuvunakool, R. Bates, A. Žídek, A. Potapenko, A. Bridgland, C. Meyer, S. A. A. Kohl, A. J. Ballard, A. Cowie, B. Romera-Paredes,

S. Nikolov, R. Jain, J. Adler, T. Back, S. Petersen, D. Reiman, E. Clancy, M. Zielinski, M. Steinegger, M. Pacholska, T. Berghammer, S. Bodenstein, D. Silver, O. Vinyals, A. W. Senior, K. Kavukcuoglu, P. Kohli, and D. Hassabis, Highly accurate protein structure prediction with AlphaFold, Nature **596**, 583 (2021).

[75] K. F. Lau and K. A. Dill, A lattice statistical mechanics model of the conformational and sequence spaces of proteins, Macromolecules **22**, 3986 (1989).

[76] A. Irbäck and C. Troein, Enumerating designing sequences in the HP model, J. Biol. Phys. **28**, 1 (2002).

[77] C. Holzgräfe, A. Irbäck, and C. Troein, Mutation-induced fold switching among lattice proteins, J. Chem. Phys. **135**, 195101 (2011).

[78] U. Bastolla, H. Frauenkron, E. Gerstner, P. Grassberger, and W. Nadler, Testing a new Monte Carlo algorithm for protein folding, Protein Eng. **32**, 52 (1998).

[79] R. Unger and J. Moult, Genetic algorithms for protein folding simulations, J. Mol. Biol. **231**, 75 (1993).

[80] S. Miyazawa and R. L. Jernigan, Estimation of effective interresidue contact energies from protein crystal structures: quasi-chemical approximation, Macromolecules **18**, 534 (1985).

[81] P. F. Faísca and K. W. Plaxco, Cooperativity and the origins of rapid, single-exponential kinetics in protein folding, Protein Sci. **15**, 1608 (2006).

[82] H. Li, C. Tang, and N. S. Wingreen, Designability of protein structures: A lattice-model study using the miyazawa-jernigan matrix, Proteins **49**, 403 (2002).

[83] S.-J. Chen, M. Hassan, R. L. Jernigan, K. Jia, D. Kihara, A. Kloczkowski, S. Kotelnikov, D. Kozakov, J. Liang, A. Liwo, S. Matysiak, J. Meller, C. Micheletti, J. C. Mitchell, S. Mondal, R. Nussinov, K. ichi Okazaki, D. Padhorny, J. Skolnick, T. R. Sosnick, G. Stan, I. Vakser, X. Zou, and G. D. Rose, Protein folds vs. protein folding: Differing questions, different challenges, Proc. Natl. Acad. Sci. USA **120**, e2214423119 (2023).

[84] A. Irbäck, C. Peterson, F. Potthast, and E. Sandelin, Design of sequences with good folding properties in coarse-grained protein models, Structure **7**, 347 (1999).

[85] A. Aina and S. Wallin, Multisequence algorithm for coarse-grained biomolecular simulations: Exploring the sequence-structure relationship of proteins, J. Chem. Phys. **147**, 095102 (2017).

[86] L. Cao, I. Goreshnik, B. Coventry, J. B. Case, L. Miller, L. Kozodoy, R. E. Chen, L. Carter, A. C. Walls, Y.-J. Park, E.-M. Strauch, L. Stewart, M. S. Diamond, D. Veesler, and D. sBaker, De novo design of picomolar SARS-CoV-2 miniprotein inhibitors, Science **370**, 426 (2020).

[87] V. K. Mulligan, H. Melo, H. I. Merritt, S. Slocum, B. D. Weitzner, A. M. Watkins, P. D. Renfrew, C. Pleasure, P. S. Arora, and R. Bonneau, Designing peptides on a quantum computer, bioRxiv , 752485 (2020).

[88] M. H. Khatami, U. C. Mendes, N. Wiebe, and P. M. Kim, Gate-based quantum computing for protein design, PLoS Comput. Biol. **19**, e1011033 (2023).

[89] V. Panizza, P. Hauke, C. Micheletti, and P. Faccioli, Protein design by integrating machine learning and quantum-encoded optimization, PRX Life **2**, 043012 (2024).

[90] M. E. Oates, P. Romero, T. Ishida, M. Ghalwash, M. J. Mizianty, B. Xue, Z. Dosztányi, V. N. Uversky, Z. Obradovic, L. Kurgan, A. K. Dunker, and J. Gough, D2P2: database of disordered protein predictions, Nucleic Acids Res. **41**, D508 (2013).

[91] Y. Shin and C. P. Brangwynne, Liquid phase condensation in cell physiology and disease, Science **357**, eaaf4382 (2017).

[92] S. Alberti and A. A. Hyman, Biomolecular condensates at the nexus of cellular stress, protein aggregation disease and ageing, Nat. Rev. Mol. Cell Biol. **18**, 196 (2021).

[93] A. Perdomo-Ortiz, N. Dickson, M. Drew-Brook, G. Rose, and A. Aspuru-Guzik, Finding low-energy conformations of lattice protein models by quantum annealing, Sci. Rep. **2**, 571 (2012).

[94] R. Babbush, A. Perdomo-Ortiz, B. O'Gorman, W. Macready, and A. Aspuru-Guzik, Construction of energy functions for lattice heteropolymer models: efficient encodings for constraint satisfaction programming and quantum annealing, *Advances in Chemical Physics: Volume 155*, Adv. Chem. Phys. **155**, 201 (2014).

[95] A. Robert, P. K. Barkoutsos, S. Woerner, and I. Tavernelli, Resource-efficient quantum algorithm for protein folding, Npj Quantum Inf. **7**, 38 (2021).

[96] H. Linn, I. Brundin, L. García-Álvarez, and G. Johansson, Resource analysis of quantum algorithms for coarse-grained protein folding models, Phys. Rev. Res. **6**, 033112 (2024).

[97] R.-H. Li, H. Doga, B. Raubenolt, S. Mostame, N. DiSanto, F. Cumbo, J. Joshi, H. Linn, M. Gaffney, A. Holden, V. Kulkarni, V. Chaudhary, K. M. M. Jr, A. A. Saki, T. Radivoyevitch, F. DiFilippo, J. Qin, O. Shehab, and D. Blankenberg, Quantum algorithm for protein structure prediction using the face-centered cubic lattice, arXiv:2507.08955 (2025).

[98] H. Linn, R.-H. Li, A. Holden, A. A. Saki, F. DiFilippo, T. Radivoyevitch, D. Blankenberg, L. García-Álvarez, and G. Johansson, Efficient quantum protein structure prediction with problem-agnostic ansatzes, arXiv:2509.18263 (2025).

[99] A. Perdomo, C. Truncik, I. Tubert-Brohman, G. Rose, and A. Aspuru-Guzik, Construction of model hamiltonians for adiabatic quantum computation and its application to finding low-energy conformations of lattice protein models, Phys. Rev. A **78**, 012320 (2008).

[100] C. Micheletti, P. Hauke, and P. Faccioli, Polymer physics by quantum computing, Phys. Rev. Lett. **127**, 080501 (2021).

[101] F. Slongo, P. Hauke, P. Faccioli, and C. Micheletti, Quantum-inspired encoding enhances stochastic sampling of soft matter systems, Sci. Adv. **9**, eadi0204 (2023).

[102] A. Irbäck, L. Knuthson, S. Mohanty, and C. Peterson, Folding lattice proteins with quantum annealing, Phys. Rev. Res. **4**, 043013 (2022).

[103] A. Irbäck, L. Knuthson, and S. Mohanty, Folding lattice proteins confined on minimal grids using a quantum-inspired encoding, Phys. Rev. E **112**, 045302 (2025).

[104] N. Chancellor, Domain wall encoding of discrete variables for quantum annealing and qaoa, Quantum Sci. Technol. **4**, 045004 (2019).

[105] Gurobi Optimization, LLC, Gurobi Optimizer Reference Manual, `https://www.gurobi.com/` (2024).

Scientific publications

In this thesis, I have addressed three lattice protein problems using methods and algortihms from two different quantum computing paradigms. The different papers either adressed a new lattice protein problem or used a different paradigm. Table 1 gives an overview to which problem I addressed and with which paradigm.

Table 1: Overview of the different lattice protein problems the papers address and if the paper used quantum annealing or gate-based quantum computing.

	quantum annealing	gate-based
Folding	Paper I and II	Paper III
Design	Paper IV	Paper V
Multi-chain	Paper VI	

Author contribution

Paper I: Folding lattice proteins with quantum annealing

Anders and I came up with the idea and I created the binary mapping. I wrote the code and performed all quantum computations, while Anders wrote and performed all classical computations. I contributed substantially to all aspects of writing the paper.

Paper II: Folding lattice proteins confined on minimal grids using a quantum-inspired encoding

Anders and I came up with the idea. I wrote the code and performed all quantum and simulated annealing computations. Sandipan and I wrote and performed the exact enumeration of all states. I contributed substantially to all aspects of writing the paper.

Paper III: Penalty-free quantum optimization applied to lattice protein folding

Stefan Prestel came up with the initial idea, while I formulated it mathematically in terms of the conflict graph. Stefan wrote the code for the divide-and-conquer computations and I wrote code used for the smaller chains. I performed all simulations and contributed substantially to all aspects of writing the paper..

Paper IV: Using quantum annealing to design lattice proteins

Me, Anders and Carsten came up with the idea and formulated the Hamiltonian and procedure. I wrote the code and performed all computations. I contributed substantially to all aspects of writing the paper.

Paper V: Designing lattice proteins with variational quantum algorithms

The idea came from everyone in the collaboration. I wrote large parts of the software. Hanna performed the computations. I contributed substantially to all aspects of writing the paper.

Paper VI: Thermodynamics of lattice protein systems using quantum annealing

Me and Anders came up with the idea. Me and Sebastian created the new mapping. I participated in writing the software for the classical parts, while Sebastian wrote it for the quantum parts. Sebastian performed most of the computations. I contributed substantially to all aspects of writing the paper.